*Information and Inference: A Journal of the IMA* (2021) **00**, 1–00
doi:10.1093/imaiai/iaaa042



# Learning a deep convolutional neural network via tensor decomposition

Samet Oymak†
*Department of Electrical and Computer Engineering, University of California, Riverside, CA 92521, USA*
†Corresponding author. Email: oymak@ece.ucr.edu

and

Mahdi Soltanolkotabi
*Ming Hsieh Department of Electrical Engineering, University of Southern California, Los Angeles, CA 90089, USA*

[Received on 8 September 2019; revised on 12 September 2020; accepted on 9 November 2020]

In this paper, we study the problem of learning the weights of a deep convolutional neural network. We consider a network where convolutions are carried out over non-overlapping patches. We develop an algorithm for simultaneously learning all the kernels from the training data. Our approach dubbed deep tensor decomposition (DeepTD) is based on a low-rank tensor decomposition. We theoretically investigate DeepTD under a realizable model for the training data where the inputs are chosen i.i.d. from a Gaussian distribution and the labels are generated according to planted convolutional kernels. We show that DeepTD is sample efficient and provably works as soon as the sample size exceeds the total number of convolutional weights in the network.

*Keywords*: tensor decomposition; convolutional neural networks; sample complexity; low-rank tensors.

## 1. Introduction

Deep neural network (DNN) architectures have led to state-of-the-art performance in many domains including image recognition, natural language processing, recommendation systems and video analysis [13, 23, 28, 48]. Convolutional neural networks (CNNs) are a class of deep, feed-forward neural networks with a specialized DNN architecture. CNNs are responsible for some of the most significant performance gains of DNN architectures. In particular, CNN architectures have led to striking performance improvements for image/object recognition tasks. CNNs, loosely inspired by the visual cortex of animals, construct increasingly higher-level features (such as mouth and nose) from lower-level features such as pixels. An added advantage of CNNs which makes them extremely attractive for large-scale applications is their remarkable efficiency which can be attributed to (1) intelligent utilization of parameters via weight sharing, (2) their convolutional nature which exploits the local spatial structure of images/videos effectively and (3) highly efficient matrix/vector multiplication involved in CNNs compared to fully connected neural network architectures.

Despite the wide empirical success of CNNs, the reason for the effectiveness of neural networks and CNNs in particular is still a mystery. Recently, there has been a surge of interest in developing more rigorous foundations for neural networks [9, 24, 29, 31, 35, 41, 42, 49, 50]. Most of this existing literature however focus on learning shallow neural networks typically consisting of zero or one hidden layer. In practical applications, depth seems to play a crucial role in constructing progressively higher-

level features from pixels. Indeed, state-of-the-art Resnet models typically have hundreds of layers. Furthermore, recent results suggest that increasing depth may substantially boost the expressive power of neural networks [11, 38].

In this paper, we propose an algorithm for approximately learning an arbitrarily deep CNN model with rigorous guarantees. Our goal is to provide theoretical insights towards better understanding when training deep CNN architectures is computationally tractable and how much data are required for successful training. We focus on a realizable model where the inputs are chosen i.i.d. from a Gaussian distribution and the labels are generated according to planted convolutional kernels. We use both labels and features in the training data to construct a tensor. Our first insight is that, in the limit of infinite data, this tensor converges to a *population* tensor which is approximately low rank and whose factors reveal the direction of the kernels. Our second insight is that even with finite data this *empirical* tensor is still approximately low rank. We show that the gap between the population and empirical tensors provably decreases with the increase in the size of the training data set and becomes negligible as soon as the size of the training data becomes proportional to the total numbers of the parameters in the planted CNN model. Combining these insights, we provide a tensor decomposition algorithm to learn the kernels from training data. We show that our algorithm approximately learns the kernels (up to sign/scale ambiguities) as soon as the size of the training data is proportional to the total number of parameters of the planted CNN model. Our results can be viewed as a step towards provable end-to-end learning of deep CNN models by establishing an algorithmic connection between low-rank tensors and CNNs. Extending the findings of earlier works [11, 24, 49], we show how tensor decomposition can be utilized to approximately learn deep networks despite the presence of nonlinearities and growing depth while requiring a near-optimal sample complexity.

## 2. Problem formulation and models

In this section, we discuss the CNN model which is the focus of this paper. A fully connected artificial neural network is composed of computational units called neurons. The neurons are decomposed into layers consisting of one input layer, one output layer and a few hidden layers with the output of each layer is fed in (as input) to the next layer. In a CNN model, the output of each layer is related to the input of the next layer by a convolution operation. In this paper, we focus on a CNN model where the stride length is equal to the length of the kernel. This is sometimes referred to as a non-overlapping convolution operation formally defined below.

DEFINITION 2.1 (Non-overlapping convolution). For two vectors $\boldsymbol{k} \in \mathbb{R}^d$ and $\boldsymbol{h} \in \mathbb{R}^{p=d\bar{p}}$, their non-overlapping convolution, denoted by $\boldsymbol{kh}$ yields a vector $\boldsymbol{u} \in \mathbb{R}^{\bar{p}=\frac{p}{d}}$ whose entries are given by

$$\boldsymbol{u}_i = \langle \boldsymbol{k}^{(\ell)}, \boldsymbol{h}[i] \rangle \quad \text{where} \quad \boldsymbol{h}[i] := \begin{bmatrix} \boldsymbol{h}_{(i-1)d+1} & \boldsymbol{h}_{(i-1)d+2} & \dots & \boldsymbol{h}_{id} \end{bmatrix}^T.$$

In this paper, it is often convenient to view convolutions as matrix/vector multiplications. This leads us to the definition of the kernel matrix below.

DEFINITION 2.2 (Kernel matrix). Consider a kernel $\boldsymbol{k} \in \mathbb{R}^d$ and any vector $\boldsymbol{h} \in \mathbb{R}^{p=\bar{p}d}$. Corresponding to the non-overlapping convolution $\boldsymbol{kh}$, we associate a kernel matrix $\boldsymbol{K} \in \mathbb{R}^{\bar{p} \times p}$ defined as $\boldsymbol{K} := \boldsymbol{I}_{\bar{p}} \otimes \boldsymbol{k}^T$. Here, $\boldsymbol{A} \otimes \boldsymbol{B}$ denotes the Kronecker product between the two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ and $\boldsymbol{I}_{\bar{p}}$ denotes the $\bar{p} \times \bar{p}$ identity matrix. We note that based on this definition $\boldsymbol{kh} = \boldsymbol{Kh}$. Throughout the paper, we shall use $\boldsymbol{K}$

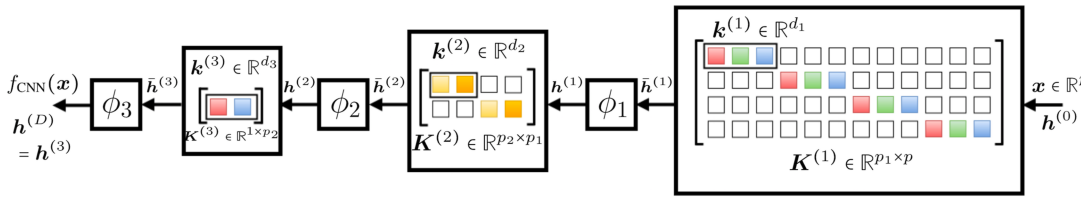



FIG. 1. Depiction of the input–output relationship of a single-filter non-overlapping CNN model along with the various notations and symbols.

interchangeably with $\boldsymbol{K}$ to denote this kernel matrix with the dependence on the underlying kernel and its non-overlapping form implied.

Remark on practical CNNs: most applications of CNNs involve computer vision where the input of a convolutional layer is a three-dimensional tensor $p_1 \times p_2 \times h$. Here $h$ is the number of feature maps (channels) and $p_1 \times p_2$ is the feature map dimension. Additionally, the kernels are also three-dimensional tensors of size $d_1 \times d_2 \times h$. For instance, at the input layer of a deep CNN, for RGB images, we have $h = 3$ for red, green and blue channels.

In this work, we shall employ one-dimensional inputs and kernels. However, the three-dimensional convolution can be collapsed into our one-dimensional setup by flattening the three-dimensional input/kernel tensors into vectors. Specifically, our one-dimensional kernel will have dimension $d = hd_1d_2$ and our one-dimensional input will have dimension $p = hp_1p_2$. Our one-dimensional input vector is obtained by flattening and concatenating the entries of the non-overlapping $d_1 \times d_2 \times h$ sub-tensors extracted from the three-dimensional input. There are $p/d = (p_1/d_1) \times (p_2/d_2)$ such sub-tensors i.e. the input tensor can be viewed as a rectangular grid with each node being a sub-tensor.

We also emphasize that, as it will be clear below, tensorized inputs/kernels are not directly related to our deep tensor decomposition (DeepTD) algorithm. The tensorized inputs/kernels are associated with individual layers and have constant order. In contrast, our tensor construction aims to capture the whole network and the order of our tensor construction is equal to the depth of the network and can be arbitrarily large.

With the definition of the non-overlapping convolution and the corresponding kernel matrix in hand we are now ready to define the CNN model which is the focus of this paper. Our CNN model is a special case of a CNN with $R$ convolutional filters at each layer. Structurally, it is obtained by a linear superposition of $R$ single-filter CNN models. For ease of exposition, the single-filter CNN input–output relationship along with the corresponding notation is depicted in Fig. 1.

• Depth and numbering of the layers. We consider a network of depth $D$ where we number the input as layer 0 and the output as layer $D$ and the hidden layers 1 to $D-1$.

• Layer dimensions and representations. We assume the input of the CNN, denoted by $\boldsymbol{x} \in \mathbb{R}^p$, consists of $p$ features and the output is a one-dimensional label. We also assume the hidden layers (numbered by $\ell = 1, 2, \ldots, D-1$) consists of $p_\ell$ units with $\bar{\boldsymbol{h}}^{(\ell)} \in \mathbb{R}^{p_\ell}$ and $\boldsymbol{h}^{(\ell)} \in \mathbb{R}^{p_\ell}$ denoting the input and output values of the units in the $\ell$th hidden layer. For consistency of our notation, we shall also define $\boldsymbol{h}^{(0)} := \boldsymbol{x} \in \mathbb{R}^p$ and note that the output of the CNN is $\boldsymbol{h}^{(D)} \in \mathbb{R}$. Also, $p_0 = p$ and $p_D = 1$.

• Kernel dimensions and representation. For $\ell = 1, \ldots D$, we assume the kernel relating the output of layer $(\ell - 1)$ to the input of layer $\ell$ is of dimension $d_\ell$ and is denoted by $\boldsymbol{k}^{(\ell)} \in \mathbb{R}^{d_\ell}$.

• Inter-layer relationship. We assume the inputs of layer $\ell$ (denoted by $\bar{\boldsymbol{h}}^{(\ell)} \in \mathbb{R}^{p_\ell}$) are related to the outputs of layer $(\ell-1)$ (denoted by $\boldsymbol{h}^{(\ell-1)} \in \mathbb{R}^{p_{\ell-1}}$) via a non-overlapping convolution

$$\bar{\boldsymbol{h}}^{(\ell)} = \boldsymbol{k}^{(\ell)} \boldsymbol{h}^{(\ell-1)} = \boldsymbol{K}^{(\ell)} \boldsymbol{h}^{(\ell-1)} \quad \text{for} \quad \ell = 1, \ldots, D.$$

In the latter equality, we have used the representation of non-overlapping convolution as a matrix/vector product involving the kernel matrix $\boldsymbol{K}^{(\ell)} \in \mathbb{R}^{p_\ell \times p_{\ell-1}}$ associated with the kernel $\boldsymbol{k}^{(\ell)} \in \mathbb{R}^{d_\ell}$ per Definition 2.2. We note that the non-overlapping nature of the convolution implies that $p_\ell = p_{\ell-1}/d_\ell$.
• Activation functions and intra-layer relationship. We assume the input of each hidden unit is related to its output by applying an activation function $\phi_\ell : \mathbb{R} \to \mathbb{R}$. More precisely, $\boldsymbol{h}^{(\ell)} := \phi_\ell(\bar{\boldsymbol{h}}^{(\ell)})$ where for a vector $\boldsymbol{u} \in \mathbb{R}^p$, $\phi_\ell(\boldsymbol{u}) \in \mathbb{R}^p$ is a vector obtained by applying the activation function $\phi_\ell$ to each of the entries of $\boldsymbol{u}$. We allow for using distinct activation functions $\{\phi_\ell\}_{\ell=1}^D$ at every layer. Throughout, we also assume all activations are 1-Lipschitz functions (i.e. $\left|\phi_\ell(a) - \phi_\ell(b)\right| \leqslant |a-b|$).
• Final output. The input–output relation of a single-filter CNN model with an input $\boldsymbol{x} \in \mathbb{R}^p$ is given by

$$\boldsymbol{x} \mapsto f_{SF}(\boldsymbol{x}) := \boldsymbol{h}^{(D)}, \tag{2.1}$$

with hidden unit relations $\boldsymbol{h}^{(\ell)} = \phi_\ell(\bar{\boldsymbol{h}}^{(\ell)})$ and $\bar{\boldsymbol{h}}^{(\ell)} = \boldsymbol{K}^{(\ell)} \boldsymbol{h}^{(\ell-1)}$. Given $R$ kernel sets $(\boldsymbol{k}^{(r,\ell)})_{\ell=1}^D$, consider $R$ single-filter CNN models

$$f_{SF,r}(\boldsymbol{x}) := \boldsymbol{h}^{(r,D)},$$

with hidden unit relations $\boldsymbol{h}^{(r,\ell)} = \phi_\ell(\bar{\boldsymbol{h}}^{(r,\ell)})$ and $\bar{\boldsymbol{h}}^{(r,\ell)} = \boldsymbol{K}^{(r,\ell)} \boldsymbol{h}^{(r,\ell-1)}$. Our model is called $R$ filter CNN and given by the linear combination of $R$ single-filter sub-networks via

$$\boldsymbol{x} \mapsto f_{CNN}(\boldsymbol{x}) := \sum_{r=1}^{R} f_{SF,r}(\boldsymbol{x}). \tag{2.2}$$

When we are working with a single-filter model, we will drop the subscript $r$ to simplify the notation. Figure 1 depicts this $R = 1$ scenario. Our general model is a special case of $R$ filter CNN where features generated by the lower layer filters (e.g. $\boldsymbol{k}^{(r,\ell)}$) are only utilized by the corresponding upper layer filters (e.g. $\boldsymbol{k}^{(r,\ell+1)}$). This structure arises from the additive form (2.2) and will help establish a connection to low-rank tensors.

## 3. Algorithm: DeepTD

This paper introduces an approach to approximating the convolutional kernels from training data based on tensor decompositions dubbed DeepTD, which consists of a carefully designed tensor decomposition. To connect these two problems, we begin by stating how we intend to construct the tensor from the training data. To this aim, given any input data $\boldsymbol{x} \in \mathbb{R}^p$, we form a $D$-way tensor $\boldsymbol{X} \in \mathbb{R}^{\bigotimes_{\ell=1}^D d_\ell}$ as follows. First, we convert $\boldsymbol{x}$ into a matrix by placing every $d_1$ consecutive entries of $\boldsymbol{x}$ as a row of a matrix of size $p_1 \times d_1$. From this matrix, we then create a 3-way tensor of size $p_2 \times d_2 \times d_1$ by grouping $d_2$ consecutive entries of each of the $d_1$ columns and so on. We repeat this procedure $D$ times to arrive at

the $D$-way tensor $\boldsymbol{X} \in \mathbb{R}^{\bigotimes_{\ell=1}^{D} d_\ell}$. We define $\mathscr{T} : \mathbb{R}^p \mapsto \mathbb{R}^{\bigotimes_{\ell=1}^{D} d_\ell}$ as the corresponding tensor operation that maps $\boldsymbol{x} \in \mathbb{R}^p$ to $\boldsymbol{X} \in \mathbb{R}^{\bigotimes_{\ell=1}^{D} d_\ell}$.

Given a set of training data consisting of $n$ input/output pairs $(\boldsymbol{x}_i, y_i) \in \mathbb{R}^p \times \mathbb{R}$, we construct a tensor $\boldsymbol{T}_n$ by tensorizing the input vectors as discussed above and calculating a weighted combination of these tensorized inputs. More precisely,

$$\boldsymbol{T}_n := \frac{1}{n}\sum_{i=1}^{n}\left(y_i - y_{avg}\right)\boldsymbol{X}_i \quad \text{where} \quad y_{avg} = \frac{1}{n}\sum_{i=1}^{n} y_i \quad \text{and} \quad \boldsymbol{X}_i = \mathscr{T}(\boldsymbol{x}_i). \tag{3.1}$$

We then perform a rank-$R$ tensor decomposition on this tensor to approximate the convolutional kernels. Specifically, we solve

$$\hat{\boldsymbol{k}}^{(r,1)}, \ldots, \hat{\boldsymbol{k}}^{(r,D)} = \underset{\boldsymbol{v}_{r,\ell} \in \mathbb{R}^{d_\ell}, \alpha_r}{\arg\min} \|\boldsymbol{T}_n - \sum_{r=1}^{R} \alpha_r \bigotimes_{\ell=1}^{D} \boldsymbol{v}_\ell\|_F^2 \text{ subject to } \left\|\boldsymbol{v}_{r,\ell}\right\|_{\ell_2} = 1 \quad \forall\, r, \ell. \tag{3.2}$$

where $\alpha_r$s are the magnitudes of the associated rank-one components. In case of a single-filter CNN model, this can be written as

$$\hat{\boldsymbol{k}}^{(1)}, \ldots, \hat{\boldsymbol{k}}^{(D)} = \underset{\boldsymbol{v}_\ell \in \mathbb{R}^{d_\ell}}{\arg\max} \left\langle \boldsymbol{T}_n, \bigotimes_{\ell=1}^{D} \boldsymbol{v}_\ell \right\rangle \quad \text{subject to} \quad \left\|\boldsymbol{v}_1\right\|_{\ell_2} = \ldots = \left\|\boldsymbol{v}_D\right\|_{\ell_2} = 1. \tag{3.3}$$

In the above, $\bigotimes_{\ell=1}^{D} \boldsymbol{v}_\ell$ denotes the tensor resulting from the outer product of $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_D$. This tensor rank decomposition is also known as CANDECOMP/PARAFAC (CP) decomposition [8] and can be solved efficiently using alternating least squares and a variety of other algorithms [2, 3, 19].[1]

At this point, it is completely unclear why the tensor $T_n$ or its rank-$R$ decomposition can yield anything useful. The main intuition is that as the data set grows ($n \to \infty$) the *empirical* tensor $\boldsymbol{T}_n$ converges close to a *population* tensor $\boldsymbol{T}$ whose rank-$R$ decomposition reveals useful information about the kernels. Specifically, we will show that

$$\lim_{n\to\infty} \boldsymbol{T}_n = \boldsymbol{T} := \mathbb{E}_{\boldsymbol{x}\sim\mathscr{N}(\boldsymbol{0},\boldsymbol{I}_p)}[f_{\text{CNN}}(\boldsymbol{x})\mathscr{T}(\boldsymbol{x})] \approx \sum_{r=1}^{R} \alpha_r \bigotimes_{\ell=1}^{D} \boldsymbol{k}^{(\ell)}, \tag{3.4}$$

with $\alpha_r$s are scalars whose value shall be discussed later on. Here, $\boldsymbol{x}$ is a Gaussian random vector with i.i.d. $\mathscr{N}(0,1)$ entries and represents a typical input with $f_{\text{CNN}}(\boldsymbol{x})$ the corresponding output and $\mathscr{T}(\boldsymbol{x})$ the tensorized input. We will also utilize a concentration argument to show that when the training data set originates from an i.i.d. distribution, for a sufficiently large training data $n$, $\boldsymbol{T}_n$ yields a good approximation of the population tensor $\boldsymbol{T}$.

[1] blackWe would like to note that while finding the best rank-$R$ TD (3.2) is NP-hard, our theoretical guarantees continue to hold when using an approximately optimal solution to (3.2). In fact, we can show that unfolding the tensor along the $i$th kernel into a $d_i \times \prod_{j\neq i} d_j$ matrix and using the top left singular vector yields a good approximation to $\boldsymbol{k}^{(i)}$. However, in our numerical simulations, we instead utilize popular software packages to solve (3.2).

We remark that while we have focused on Gaussian input distributions, the results can be easily extended to other distributions by using an associated score function. In particular, if the data points have a distribution $p(\cdot) : \mathbb{R}^p \to \mathbb{R}$, the score function is given by $\mathscr{S}(\boldsymbol{x}) = -\frac{\nabla p(\boldsymbol{x})}{p(\boldsymbol{x})}$. In this case, following Stein's identity [44], DeepTD can still be used after setting $\boldsymbol{X}_i$ to $\mathscr{T}(\mathscr{S}(\boldsymbol{x}_i))$ in (3.1). Developing related theoretical guarantees for other distributions using this strategy is left to future work.

Another perhaps perplexing aspect of the construction of $\boldsymbol{T}_n$ in (3.1) is the subtraction by $y_{avg}$ in the weights. The reason this may be a source of confusion is that based on the intuition above

$$\mathbb{E}[\boldsymbol{T}] = \mathbb{E}\Big[\frac{1}{n}\sum_{i=1}^{n} y_i \boldsymbol{X}_i\Big] = \mathbb{E}\Big[\frac{1}{n-1}\sum_{i=1}^{n}\Big(y_i - y_{avg}\Big)\boldsymbol{X}_i\Big] = \frac{n}{n-1}\,\mathbb{E}[\boldsymbol{T}_n] \approx \sum_{r=1}^{R}\alpha_r \bigotimes_{\ell=1}^{D} \boldsymbol{k}^{(\ell)},$$

so that the subtraction by the average seems completely redundant. The main purpose of this subtraction is to ensure the weights $y_i - y_{avg}$ are centered (have mean zero). This centering allows for a much better concentration of the empirical tensor around its population counter part and is crucial to the success of our approach, in particular ensuring sample efficiency. The centering achieves a variance reduction affect by removing the redundant label mean. Note that the label mean is not informative for our algorithm because in expectation it has no contribution to the tensor construction due to zero-mean input features.

We emphasize that, besides labels, centering input features can be beneficial as well. However, this work assumes zero-mean inputs (which are already centered); thus, we opted not to include this procedure. We would like to point out that such a centering procedure is reminiscent of batch-normalization heuristics deployed when training DNNs.

Finally, we note that based on (3.4), the rank-$R$ tensor decomposition step can recover the convolutional kernels $\{\boldsymbol{k}^{(r,\ell)}\}_{r,\ell=1,1}^{R,D}$ up to sign and scaling ambiguities. Unfortunately, depending on the activation function, it may be impossible to overcome these ambiguities. For instance, if the activations are homogeneous (i.e. $\phi_\ell(ax) = a\phi_\ell(x)$), then scaling up one layer and scaling down the other layer by the same amount does not change the overall function $f_{\text{CNN}}(\cdot)$. Similarly, if the activations are odd functions, negating two of the layers at the same time preserves the overall function. In Appendix A, we discuss some heuristics and theoretical guarantees for overcoming these sign/scale ambiguities.

## 4. Main results

In this section, we introduce our theoretical results for DeepTD. We will discuss these results in three sections. In Section 4.1, we show that the empirical tensor concentrates around its population counterpart. Then, in Section 4.2, we show that the population tensor is well-approximated by a rank-$R$ tensor whose factors reveal the convolutional kernels. Finally, in Section 4.3, we combine these results to show DeepTD can approximately learn the convolutional kernels up to sign/scale ambiguities.

### 4.1 *Concentration of the empirical tensor*

Our first result shows that the empirical tensor concentrations around the population tensor. We measure the quality of this concentration via the tensor spectral norm defined below.

DEFINITION 4.1 The spectral norm of a tensor $\boldsymbol{X} \in \mathbb{R}^{\bigotimes_{\ell=1}^{D} d_\ell}$ is given by the supremum $|||\boldsymbol{T}||| = \sup_{\|\boldsymbol{v}_\ell\|_{\ell_2}\leqslant 1}\Big\langle \boldsymbol{T}, \bigotimes_{\ell=1}^{D}\boldsymbol{v}_\ell\Big\rangle$.

THEOREM 4.2 Consider a CNN model $\boldsymbol{x} \mapsto f_{CNN}(\boldsymbol{x})$ of the form (2.2) consisting of $D \geqslant 2$ layers with unit Euclidian norm convolutional kernels $(\boldsymbol{k}^{(r,\ell)})_{r,\ell=1,1}^{R,D}$. Let $\boldsymbol{x} \in \mathbb{R}^p$ be a Gaussian random vector distributed as $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_p)$ with the corresponding labels $y = f_{CNN}(\boldsymbol{x})$ generated by the CNN model and $\boldsymbol{X} := \mathscr{T}(\boldsymbol{x})$ the corresponding tensorized input. Suppose the data set consists of $n$ training samples where the feature vectors $\boldsymbol{x}_i \in \mathbb{R}^p$ are distributed i.i.d. $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_p)$ with the corresponding labels $y_i = f_{CNN}(\boldsymbol{x}_i)$ generated by the same CNN model and $\boldsymbol{X}_i := \mathscr{T}(\boldsymbol{x}_i)$ the corresponding tensorized input. Suppose $n \geqslant (\sum_{\ell=1}^{D} d_\ell) \log D$. Then the empirical tensor $\boldsymbol{T}_n$ and population tensor $\boldsymbol{T}$ defined based on this data set obey

$$\left|\!\left|\!\left| \boldsymbol{T}_n - \boldsymbol{T} \right|\!\right|\!\right| := \left|\!\left|\!\left| \frac{1}{n} \sum_{i=1}^{n} (y_i - y_{\text{avg}}) \boldsymbol{X}_i - \mathbb{E}[y\boldsymbol{X}] \right|\!\right|\!\right| \leqslant cR \frac{\sqrt{\left(\sum_{\ell=1}^{D} d_\ell\right) \log D} + t}{\sqrt{n}}, \tag{4.1}$$

with probability at least $1 - 5e^{-\min(t^2, t\sqrt{n}, n)}$, where $c > 0$ is an absolute constant.

The theorem above shows that the empirical tensor approximates the population tensor with high probability. Our convolutional model has $RD$ kernels and the total degrees of freedom is $\text{DoF} = R \sum_{\ell=1}^{D} d_\ell$. The quality of this approximation grows with the square root of degrees of freedom and is inversely proportional to the square root of the number of samples ($n$) which are typical scalings in statistical learning. We achieve small approximation error as soon as

$$n \gtrsim R \log D \times \text{DoF},$$

which means our results achieve optimal sample complexity up to a factor of $R \log D$ and the approximation error is less than $\epsilon$ as soon as the number of observations exceeds the number of parameters in the model by $R \log D$ i.e. $n \gtrsim R^2 (\sum_{\ell=1}^{D} d_\ell) \frac{\log D}{\epsilon^2}$.

This result is a special case of Theorem 8.4 which allows for arbitrary kernel lengths in which case $R$ on the right-hand side is replaced by the Lipschitz constant of the network $\prod_{\ell=1}^{D} \left\| \boldsymbol{k}^{(\ell)} \right\|_{\ell_2}$. As we will see in the next sections, $\boldsymbol{T}$ is approximately rank-$R$ and its tensor spectral norm $|\!|\!|\boldsymbol{T}|\!|\!|$ is roughly on the order of $\max_{r=1}^{R} \prod_{\ell=1}^{D} \left\| \boldsymbol{k}^{(r,\ell)} \right\|_{\ell_2}$ to $\prod_{\ell=1}^{D} \left\| \boldsymbol{k}^{(\ell)} \right\|_{\ell_2}$ (assuming its rank one summands do not cancel each other out). Hence, our general result looks identical to (4.1) after scaling and we achieve $\left|\!\left|\!\left| \boldsymbol{T}_n - \boldsymbol{T} \right|\!\right|\!\right| / |\!|\!|\boldsymbol{T}|\!|\!| \lesssim c' R \sqrt{\frac{\left(\sum_{\ell=1}^{D} d_\ell\right) \log D}{n}}$.

### 4.2 *Low-rank approximation of the population tensor*

Our second result shows that the population tensor can be approximated by a rank-$R$ tensor. To explain the structure of this rank-$R$ tensor and quantify the quality of this approximation, we require a few definitions. The first quantity roughly captures the average amount by which the nonlinear activations amplify or attenuate the size of an input feature at the output.

DEFINITION 4.3 (CNN gain). Let $\boldsymbol{x} \sim \mathcal{N}(0, \boldsymbol{I}_p)$ and define the hidden unit/output values of the CNN based on this random input per equations (2.2). We define the CNN gain associated with $r$th single-filter sub-network as $\alpha_{r,\text{CNN}} = \prod_{\ell=1}^{D} \mathbb{E}[\phi'_\ell(\bar{\boldsymbol{h}}_1^{(r,\ell)})]$. In words, this is the product of expectations of the activations evaluated at the first entry of each layer. For non-differentiable activations, $\phi'_\ell$ should be interpreted as the average of the left and right derivatives. For instance, when $\phi_\ell(z) = \text{ReLU}(z)$ then $\phi'_\ell(0) = 1/2$.

This quantity is the product of the average slopes of the activations evaluated along a path connecting the first input feature to the first hidden units across the layers all the way to the output. We note that this quantity is the same when calculated along any path connecting an input feature to the output passing through the hidden units. Therefore, this quantity can be thought of as the average gain (amplification or attenuation) of a given input feature due to the nonlinear activations in the network. To gain some intuition, consider a ReLU network, which is mostly inactive. Then the network is dead and $\alpha_{\text{CNN}} \approx 0$. On the other extreme, if all ReLU units are active the network operates in the linear regime and $\alpha_{\text{CNN}} = 1$. We would like to point out that $\alpha_{\text{CNN}}$ can in many cases be bounded from below by a constant. For instance, as proven in Section 9, for ReLU activations as long as the kernels obey

$$\left(\mathbf{1}^T \boldsymbol{k}^{(r,\ell)}\right) \geqslant 4 \left\| \boldsymbol{k}^{(r,\ell)} \right\|_{\ell_2}, \tag{4.2}$$

then $\alpha_{\text{CNN}} \geqslant 1/4$. Another example is the softplus activation $\phi_\ell(x) = \log(1 + e^x)$ for which we prove $\alpha_{\text{CNN}} \geqslant 0.3$ under similar assumptions (also in Section 9). We note that an assumption similar to (4.2) is needed for the network to be active. This is because if the kernel sums are negative one can show that with high probability, all the ReLUs after the first layer will be inactive and the network will be dead. With this definition in hand, we are now ready to describe the form of the rank-$R$ tensor that approximates the population tensor.

DEFINITION 4.4 (Rank-$R$ CNN tensor). We define the rank-$R$ CNN tensor $\boldsymbol{L}_{\text{CNN}} \in \mathbb{R}^{\bigotimes_{\ell=1}^{D} d_\ell}$ as $\boldsymbol{L}_{\text{CNN}} = \sum_{r=1}^{R} \alpha_{r,\text{CNN}} \bigotimes_{\ell=1}^{D} \boldsymbol{k}^{(r,\ell)}$. That is, the product of the kernels $\{\boldsymbol{k}^{(r,\ell)}\}_{\ell=1}^{D}$ scaled by the CNN gain $\alpha_{r,\text{CNN}}$.

To quantify how well the rank-$R$ CNN tensor approximates the population tensor, we need two definitions. The first definition concerns the activation functions.

DEFINITION 4.5 (Activation smoothness). We assume the activations are differentiable everywhere and $S$-smooth (i.e. $|\phi_\ell'(x) - \phi_\ell'(y)| \leqslant S|x - y|$ for all $x, y \in \mathbb{R}$) for some $S \geqslant 0$.

The reason smoothness of the activations plays a role in the quality of the rank-$R$ approximation is that smoother activations translate into smoother variations in the entries of the population tensor. Therefore, the population tensor can be better approximated by a low-rank tensor. The second definition captures how diffused the kernels are.

DEFINITION 4.6 (Kernel diffuseness parameter). Given kernels $\{\boldsymbol{k}^{(r,\ell)}\}_{\ell=1}^{D}$ with dimensions $\{d_\ell\}_{\ell=1}^{D}$, the kernel diffuseness parameter $\mu$ is defined as $\mu = \sup_{1 \leqslant \ell \leqslant D} \sqrt{d_\ell} \left\| \boldsymbol{k}^{(r,\ell)} \right\|_{\ell_\infty} / \left\| \boldsymbol{k}^{(r,\ell)} \right\|_{\ell_2}$.

The less diffused (or more spiky) the kernels are, the more the population tensor fluctuates and thus the quality of the approximation to a rank-$R$ tensor decreases. With these definitions in place, we are now ready to state our theorem on approximating a population tensor with a rank-$R$ tensor.

THEOREM 4.7 Consider the setup of Theorem 4.2. Also, assume the activations are $S$-smooth per Definition 4.5 and the convolutional kernels are $\mu$-diffused per Definition 4.6. Then, the population tensor $\boldsymbol{T} := \mathbb{E}[y\boldsymbol{X}]$ can be approximated by the rank-$R$ tensor $\boldsymbol{L}_{\text{CNN}} := \sum_{r=1}^{R} \alpha_{r,\text{CNN}} \bigotimes_{\ell=1}^{D} \boldsymbol{k}^{(r,\ell)}$ as follows

$$\|\boldsymbol{T} - \boldsymbol{L}_{\text{CNN}}\|_F \leqslant \sqrt{8\pi}\, \mu S \frac{RD}{\sqrt{\min_\ell d_\ell}}. \tag{4.3}$$

The theorem above is a special case of Theorem 8.5 which allows for arbitrary kernel norms. The theorem above states that the quality of the rank-$R$ approximation deteriorates with increase in the smoothness of the activations and the diffuseness of the convolutional kernels. As mentioned earlier increase in these parameters leads to more fluctuations in the population tensor making it less likely that it can be well approximated by a rank-$R$ tensor. Recalling low-rank CNN tensor of Definition 4.4, assuming its rank one summands are uncorrelated, we have that

$$\|\boldsymbol{L}_{\mathrm{CNN}}\|_F \approx \sqrt{\sum_{1\leqslant r\leqslant R} \alpha_{r,\mathrm{CNN}}^2 \prod_{\ell=1}^{D} \left\|\boldsymbol{k}^{(r,\ell)}\right\|_{\ell_2}} = \sqrt{\sum_{1\leqslant r\leqslant R} \alpha_{r,\mathrm{CNN}}^2} := \sqrt{R}\alpha_{\mathrm{CNN}}.$$

and therefore the relative Frobenius error in this approximation is bounded by

$$\frac{\|\boldsymbol{T}-\boldsymbol{L}_{\mathrm{CNN}}\|_F}{\|\boldsymbol{L}_{\mathrm{CNN}}\|_F} \lesssim \sqrt{8\pi}\frac{\mu S}{\alpha_{\mathrm{CNN}}}\frac{\sqrt{R}D}{\sqrt{\min_{\ell} d_\ell}}.$$

We would like to note that for many activations the smoothness is bounded by a constant. For instance, for the softplus activation ($\phi(x) = \log(1+e^x)$) and one can show that $S \leqslant 1$. As stated earlier, under appropriate assumptions on the kernels and activations, the CNN gain $\alpha_{\mathrm{CNN}}$ is also bounded from below by a constant. Assuming kernels are sufficiently diffused so that the diffuseness parameter is bounded by a constant, we can then conclude that $\frac{\|\boldsymbol{T}-\boldsymbol{L}_{\mathrm{CNN}}\|_F}{\|\boldsymbol{L}_{\mathrm{CNN}}\|_F} \lesssim \frac{c}{\alpha_{\mathrm{CNN}}}\frac{\sqrt{R}D}{\sqrt{\min_{\ell} d_\ell}}$. This implies that as soon as the length of the convolutional patches scale with the square of total number of filters ($RD^2$) of the network approximation is sufficiently good.

Our back-of-the-envelope calculations suggest that the proposed scaling can be improved. In particular, our proof strategy does not take advantage of the possible low-correlations of the errors arising from different sub-networks and different layers, instead adds them up. For instance, we believe incoherence of the approximation errors associated with rank 1 summands should make the relative error independent of $R$. Sharpening the analysis to achieve the correct scaling is an interesting future research direction. Finally, we would like to note that while we have assumed differentiable and smooth activations we expect our results to apply to popular non-differentiable activations such as ReLU activations.

### 4.3 *Learning the convolutional kernels*

Merging Theorems 4.2 and 4.7 provides an immediate corollary on finite sample approximation to low-rank tensor obtained by CNN filters.

THEOREM 4.8 (Main theorem—low-rank CNN). Consider the setups of Theorems 4.2 and 4.7. Assume activations are $S$-smooth and the convolutional kernels are $\mu$-diffused per Definitions 4.5 and 4.6. Suppose the ground truth kernels $(\boldsymbol{k}^{(r,\ell)})_{r,\ell=(1,1)}^{R,D}$ have unit Euclidian norm. Then, the empirical tensor $\boldsymbol{T}_n$ obeys

$$\left|\left|\left|\boldsymbol{T}_n - \boldsymbol{L}_{\mathrm{CNN}}\right|\right|\right| \leqslant cR\frac{\sqrt{\left(\sum_{\ell=1}^{D} d_\ell\right)\log D} + t}{\sqrt{n}} + \sqrt{8\pi}R\mu S\frac{D}{\sqrt{\min_{\ell} d_\ell}}.$$

Now that we have a tensor spectral norm approximation guarantee, one can use any robustness result on low-rank tensor decomposition to guarantee that CNN filters $\boldsymbol{k}^{(r,\ell)}$ can be recovered from an application of low-rank tensor decomposition. Unlike matrices, low-rank tensors can have highly coherent rank one summands which makes it more challenging to provide a insightful perturbation analysis. As a result, we will focus on the $R = 1$ case to state a clean result. The next theorem guarantees that output of tensor decomposition has high correlation to the ground truth kernels to provide finite sample guarantees for DeepTD.

THEOREM 4.9 (Main theorem—single filter). Consider a single-filter CNN model ($R = 1$) with convolutional kernels $(\boldsymbol{k}^{(\ell)})_{\ell=1}^{D}$. Suppose $n \geqslant (\sum_{\ell=1}^{D} d_\ell) \log D$. Assume activations are $S$-smooth per Definition 4.5 and the convolutional kernels are $\mu$-diffused per Definition 4.6. The DeepTD estimates of the convolutional kernels given by (3.2) using the empirical tensor $\boldsymbol{T}_n$ obeys

$$\frac{\prod_{\ell=1}^{D} |\left\langle \boldsymbol{k}^{(\ell)}, \hat{\boldsymbol{k}}^{(\ell)} \right\rangle|}{\prod_{\ell=1}^{D} \left\| \boldsymbol{k}^{(\ell)} \right\|_{\ell_2}} \geqslant 1 - \frac{2}{\alpha_{\mathrm{CNN}}} \left( c \frac{\sqrt{\left(\sum_{\ell=1}^{D} d_\ell\right) \log D} + t}{\sqrt{n}} + \sqrt{8\pi}\mu S \cdot \sup_{\ell} \prod_{i=1}^{\ell} \left\| \boldsymbol{k}^{(i)} \right\|_{\ell_2} \frac{D}{\sqrt{\min_{\ell} d_\ell}} \right),$$

with probability at least $1 - 5e^{-\min(t^2, t\sqrt{n}, n)}$, where $c > 0$ is an absolute constant.

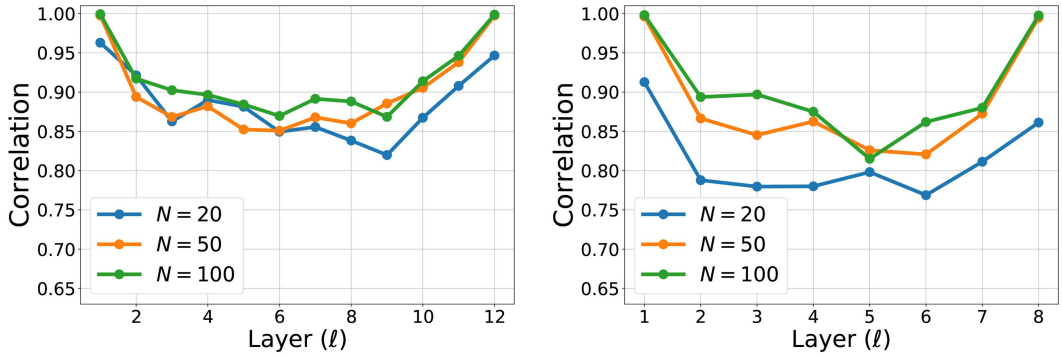
The above theorem is our main result on learning a non-overlapping CNN with a single kernel at each layer. It demonstrates that estimates $\hat{\boldsymbol{k}}^{(\ell)}$ obtained by DeepTD have significant inner product with the ground truth kernels $\boldsymbol{k}^{(\ell)}$ with high probability, using only few samples. Indeed, similar to the discussion after Theorem 4.7 assuming the activations are sufficiently smooth and the convolutional kernels are unit norm and sufficiently diffused, the theorem above can be simplified as follows:

$$\frac{\prod_{\ell=1}^{D} |\left\langle \boldsymbol{k}^{(\ell)}, \hat{\boldsymbol{k}}^{(\ell)} \right\rangle|}{\prod_{\ell=1}^{D} \left\| \boldsymbol{k}^{(\ell)} \right\|_{\ell_2}} \geqslant 1 - c \left( \frac{\sqrt{\left(\sum_{\ell=1}^{D} d_\ell\right) \log D}}{\sqrt{n}} + \frac{D}{\sqrt{\min_{\ell} d_\ell}} \right).$$

Thus, the kernel estimates obtained via DeepTD are well aligned with the true kernels as soon as the number of samples scales with the total number of parameters in the model and the length of the convolutional kernels (i.e. the size of the batches) scales quadratically with the depth of the network. Note that our theorem guarantees a large correlation (in absolute value) and do not recover signs and scalings of the ground truth kernels. Further theory and algorithms on resolving sign and scaling ambiguities are provided in Appendix A which also sheds light on the role of centering trick (subtracting the mean of the labels) for faster learning.

## 5. Numerical experiments

Our goal in this section is to numerically corroborate the theoretical predictions of Section 4. To this aim, we use a single-filter CNN model of the form (2.2) with $D$ layers and ReLU activations and set the kernel lengths to be all equal to each other i.e. $d_4 = \ldots = d_1 = d$. We use the identity activation for the last layer (i.e. $\phi_D(z) = z$) with the exception of the last experiment where we use a ReLU activation (i.e. $\phi_D(z) = \max(0, z)$). We conducted our experiments in Python using the Tensorly library for the tensor




(a) $d = 2, D = 12$. (b) $d = 3, D = 8$.

FIG. 2. Correlations ($\text{corr}(\hat{\boldsymbol{k}}^{(\ell)}, \boldsymbol{k}^{(\ell)}) = \left|\left\langle \hat{\boldsymbol{k}}^{(\ell)}, \boldsymbol{k}^{(\ell)} \right\rangle\right|$) between the DeepTD estimate and the ground truth kernels for different layers and various oversampling ratios ($N = \frac{n}{\sum_{\ell=1}^{D} d_\ell}$).

decomposition in DeepTD [27]. Each curve in every figure is obtained by averaging 100 independent realizations of the same CNN learning procedure. Similar to our theory, we use Gaussian data points $\boldsymbol{x}$ and ground truth labels $\boldsymbol{y} = f_{\text{CNN}}(\boldsymbol{x})$.

We conduct two sets of experiments: the first set focuses on larger values of depth $D$ and the second set focuses on larger values of width $d$. In all experiments, kernels are generated with random Gaussian entries and are normalized to have unit Euclidean norm. For the ReLU activation, if one of the kernels have all negative entries, the output is trivially zero and learning is not feasible. To address this, we consider operational networks where at least 50% of the training labels are nonzero. Here, the number 50% is arbitrarily chosen and we verified that similar results hold for other values.

Finally, to study the effect of finite samples, we let the sample size grow proportional to the total degrees of freedom $\sum_{\ell=1}^{D} d_\ell$. In particular, we set an oversampling factor $N = \frac{n}{\sum_{\ell=1}^{D} d_\ell}$ and carry out the experiments for $N \in \{10, 20, 50, 100\}$. While our theory requires $N \gtrsim \log D$, in our experiments, we typically observe that improvement is marginal after $N = 50$.

In Fig. 2, we consider two networks with $d = 2, D = 12$ and $d = 3, D = 8$ configurations. We plot the absolute correlation between the ground truth and the estimates as a function of layer depth. For each hidden layer $1 \leqslant \ell \leqslant D$, our correlation measure ($y$-axis) is

$$\text{corr}(\hat{\boldsymbol{k}}^{(\ell)}, \boldsymbol{k}^{(\ell)}) = \left|\left\langle \hat{\boldsymbol{k}}^{(\ell)}, \boldsymbol{k}^{(\ell)} \right\rangle\right|.$$

This number is between 0 and 1 as the kernels and their estimates both have unit norm. We observe that for both $d = 2$ and $d = 3$, DeepTD consistently achieves correlation values above 75% for $N = 20$. While our theory requires $d$ to scale quadratically with depth i.e. $d \gtrsim D^2$, we find that even small $d$ values work well in our experiments. The effect of sample size becomes evident by comparing $N = 20$ and $N = 50$ for the input and output layers ($\ell = 1, \ell = D$). In this case, $N = 50$ achieves perfect correlation. Interestingly, correlation values are smallest in the middle layers. In fact, this even holds when $N$ is large suggesting that the rank one approximation of the population tensor provides worst estimates for the middle layers.

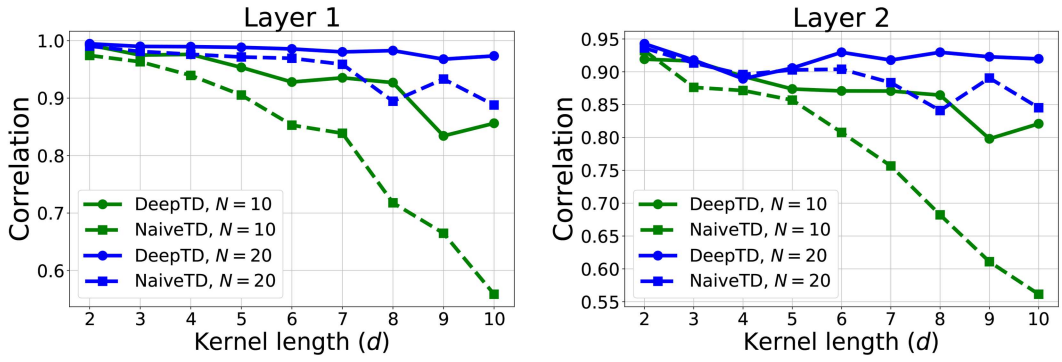



FIG. 3. DeepTD estimate vs. NaiveTD estimate when final activation is ReLU. Bias of NaiveTD results in significantly worse performance.

In Fig. 3, we use a ReLU activation in the final layer and assess the impact of the centering procedure of the DeepTD algorithm which is a major theme throughout the paper. We define the NaiveTD algorithm which solves (3.2) without centering in the empirical tensor i.e.

$$\hat{\boldsymbol{k}}^{(1)}, \ldots, \hat{\boldsymbol{k}}^{(D)} = \underset{\boldsymbol{v}_\ell \in \mathbb{R}^{d_\ell}}{\arg\max} \left\langle \frac{1}{n} \sum_{i=1}^{n} y_i \boldsymbol{X}_i, \bigotimes_{\ell=1}^{D} \boldsymbol{v}_\ell \right\rangle \quad \text{subject to} \quad \|\boldsymbol{v}_1\|_{\ell_2} = \ldots = \|\boldsymbol{v}_D\|_{\ell_2} = 1. \tag{5.1}$$

Since the activation of the final layer is ReLU, the output has a clear positive bias in expectation which will help demonstrating the importance of centering. We find that for smaller oversampling factors of $N = 10$ or $N = 20$, DeepTD has a visibly better performance compared with NaiveTD. The correlation difference is persistent among different layers (we plotted only Layers 1 and 2) and appears to grow with increase in the kernel size $d$.

Finally, in Fig. 4, we assess the impact of activation nonlinearity by comparing the ReLU and identity activations in the final layer. We plot the first and final layer correlations for this setup. While the correlation performances of the first layer are essentially identical, the ReLU activation (dashed lines) achieves significantly lower correlation at the final layer. This is not surprising as the final layer passes through an additional nonlinearity.

## 6. Related work

Our work is closely related to the recent line of papers on neural networks as well as tensor decompositions. We briefly discuss this related literature.

Neural networks: learning neural networks is a non-trivial task involving nonlinearities and non-convexities. Consequently, existing theory works consider different algorithms, network structures and assumptions. A series of recent work focus on learning zero or one-hidden layer fully connected neural networks with random inputs and planted models [18, 20, 21, 31, 35, 41, 50]. Other publications [9, 15, 17, 22, 49] consider the problem of learning a CNN with 1-hidden layer. In particular, first three of these focus on learning non-overlapping CNNs as in this paper (albeit in the limit of infinite training data). The papers above either focus on characterizing the optimization landscape, or population landscape, or

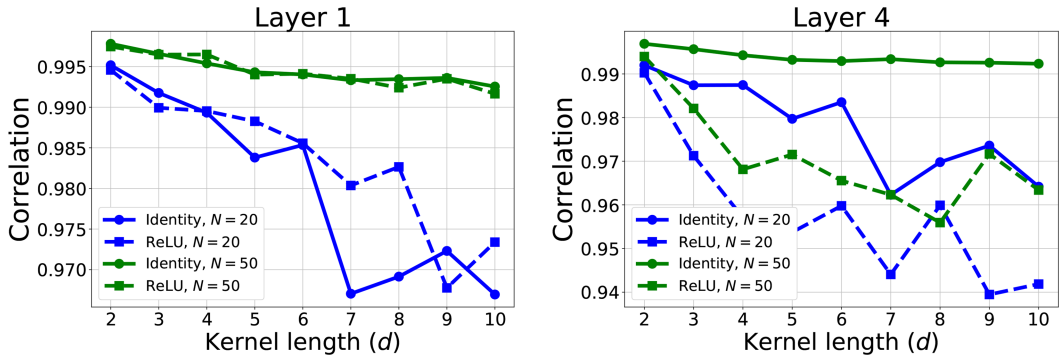



FIG. 4. Comparison of performance of DeepTD when the final activation is ReLU in lieu of the identity activation.

providing exact convergence guarantees for gradient descent. In comparison, in this paper we focus on approximate convergence guarantees using tensor decompositions for arbitrary deep networks. A few recent publications [1, 7, 10, 16, 25, 36, 37, 39, 42, 43] consider the training problem when the network is over-parametrized and study the over-fitting ability of such networks. Baldi & Vershynin and others [5, 6, 33, 34] study the capacity and generalization ability of deep networks. Closer to our work, [30] and [4] consider provable algorithms for deep networks using layer-wise algorithms. In comparison to our work, [30] applies to a very specific generative model that assumes a discrete data distribution and studies the population loss in lieu of the empirical loss. Arora *et al.* [4] study deep models but use activation functions that are not commonly used and assume random network weights. In comparison, we work with realistic activations and arbitrary network weights.

Tensor decomposition: tensors are powerful tools to model a wide variety of big-data problems [2, 40]. Recent years have witnessed a growing interest in tensor decomposition techniques to extract useful latent information from the data [2, 19]. The connection between tensors and neural networks has been noticed by several papers [11, 12, 24, 26, 32]. Cohen *et al.* [11, 12] relate CNNs and tensor decompositions to provide insights on the expressivity of CNNs. Mondelli & Montanari [32] connect the hardness of learning shallow networks to tensor decompositions. Closer to this paper, [24] and [50] apply tensor decomposition on *one*-hidden layer, fully connected networks to approximately learn the latent weight matrices.

## 7. Conclusion

In this paper, we studied a multilayer CNN model with depth $D$. Our model is a special case of a CNN with $R$ filter at each layer. We establish a connection between approximating the CNN kernels and higher order tensor decompositions. Based on this, we proposed an algorithm for simultaneously learning all kernels called the DeepTD. This algorithm builds a $D$-way tensor based on the training data and applies a rank-$R$ tensor factorization algorithm to this tensor to simultaneously estimate all of the convolutional kernels. Assuming the input data are distributed i.i.d. according to a Gaussian model with corresponding output generated by a planted set of convolutional kernels, we prove DeepTD can approximately learn all kernels with a near minimal amount of training data. A variety of numerical experiments complement our theoretical findings.

## 8. Proofs of the main theorems

In this section, we will prove our main results. Throughout, for a random variable $X$, we use $\mathbf{zm}(X)$ to denote $X - \mathbb{E}[X]$. Simply stated, $\mathbf{zm}(X)$ is the centered version of $X$. Let $\mathcal{RO}$ be the set of rank-one tensors $\bigotimes_{\ell=1}^{D} \boldsymbol{v}_\ell$ satisfying $\|\boldsymbol{v}_i\|_{\ell_2} \leqslant 1$ for all $1 \leqslant i \leqslant D$. For a random vector/matrix/tensor $\boldsymbol{X}$, $\mathbf{zm}(\boldsymbol{X})$ denotes the vector/matrix/tensor obtained by applying the $\mathbf{zm}()$ operation to each entry. For a tensor $\boldsymbol{T}$, we use $\|\boldsymbol{T}\|_F$ to denote the square root of the sum of squares of the entries of the tensor. Stated differently, this is the Euclidean norm of a vector obtained by rearranging the entries of the tensor. Throughout, we use $c$, $c_1$, $c_2$ and $C$ to denote fixed numerical constants whose values may change from line to line. We begin with some useful definitions and lemmas.

### 8.1 *Useful concentration lemmas and definitions*

In this section, we gather some useful definitions and well-known lemmas that will be used frequently throughout our concentration arguments.

DEFINITION 8.1 (Orlicz norms). For a scalar random variable, Orlicz-$a$ norm is defined as

$$\|X\|_{\psi_a} = \sup_{k \geqslant 1} k^{-1/a} (\mathbb{E}[|X|^k])^{1/k}$$

Orlicz-$a$ norm of a vector $\boldsymbol{x} \in \mathbb{R}^p$ is defined as $\|\boldsymbol{x}\|_{\psi_a} = \sup_{\boldsymbol{v} \in \mathscr{B}^p} \|\boldsymbol{v}^T \boldsymbol{x}\|_{\psi_a}$ where $\mathscr{B}^p$ is the unit $\ell_2$ ball. The sub-exponential norm is the function $\|\cdot\|_{\psi_1}$ and the subgaussian norm the function $\|\cdot\|_{\psi_2}$.

We now state a few well-known results that we will use throughout the proofs. This results are standard and are stated for the sake of completeness. The first lemma states that the product of subgaussian random variables are sub-exponential.

LEMMA 8.1 Let $X, Y$ be subgaussian random variables. Then $\|XY\|_{\psi_1} \leqslant \|X\|_{\psi_2} \|Y\|_{\psi_2}$.

The next lemma connects Orlicz norms of sum of random variables to the sum of the Orlicz norm of each random variable.

LEMMA 8.2 Suppose $X, Y$ are random variables with bounded $\|\cdot\|_{\psi_a}$ norm. Then $\|X + Y\|_{\psi_a} \leqslant 2 \max\{\|X\|_{\psi_a}, \|Y\|_{\psi_a}\}$. In particular, $\|X - \mathbb{E}\, X]\|_{\psi_a} \leqslant 2\|X\|_{\psi_a}$.

The lemma below can be easily obtained by combining the previous two lemmas.

LEMMA 8.3 Let $X, Y$ be subgaussian random variables. Then $\|\mathbf{zm}(XY)\|_{\psi_1} \leqslant 2\|X\|_{\psi_2} \|Y\|_{\psi_2}$.

Finally, we need a few standard chaining definitions.

DEFINITION 8.2 (Admissible sequence [46]). Given a set $T$ an admissible sequence is an increasing sequence $\{A_n\}_{n=0}^{\infty}$ of partitions of $T$ such that $|A_n| \leqslant N_n$ where $N_0 = 1$ and $N_n = 2^{2^n}$ for all $n \geqslant 1$.

For the following discussion, $\Delta_d(A_n(t))$ will be the diameter of the set $S \in A_n$ that contains $t$, with respect to the $d$ metric.

DEFINITION 8.3 ($\gamma_a$ functional [46]). Given $a > 0$ and a metric space $(T, d)$, we define

$$\gamma_a(T, d) = \inf \sup_{t \in T} \sum_{n \geqslant 0} 2^{n/a} \Delta_d(A_n(t)),$$

where the infimum is taken over all admissible sequences.

The following lemma upper bounds $\gamma_\alpha$ functional with covering numbers of $T$. The reader is referred to [45, Section 1.2], [14, Equation (2.3)] and [35, Lemma $D$.17].

LEMMA 8.4 (Dudley's entropy integral). Let $N(\varepsilon)$ be the $\varepsilon$ covering number of the set $T$ with respect to the $d$ metric. Then

$$\gamma_\alpha(T,d) \leqslant C_\alpha \int_0^\infty \log^{1/\alpha} N(\varepsilon)\,\mathrm{d}\varepsilon,$$

where $C_\alpha > 0$ depends only on $\alpha > 0$.

## 8.2 *Concentration of the empirical tensor (Proof of Theorem 4.2)*

We will prove the following general version of Theorem 4.2 which allows for arbitrary kernel norms.

THEOREM 8.4 Consider a CNN model $\boldsymbol{x} \mapsto f_{CNN}(\boldsymbol{x})$ of the form (2.2) consisting of $D \geqslant 2$ layers with convolutional kernels $\boldsymbol{k}^{(1)}, \boldsymbol{k}^{(2)}, \ldots, \boldsymbol{k}^{(D)}$ of lengths $d_1, d_2, \ldots, d_D$. Let $\boldsymbol{x} \in \mathbb{R}^p$ be a Gaussian random vector distributed as $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_p)$ with the corresponding labels $y = f_{CNN}(\boldsymbol{x})$ generated by the CNN model and $\boldsymbol{X} := \mathscr{T}(\boldsymbol{x})$ the corresponding tensorized input. Suppose the data set consists of $n$ training samples where the feature vectors $\boldsymbol{x}_i \in \mathbb{R}^p$ are distributed i.i.d. $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_p)$ with the corresponding labels $y_i = f_{CNN}(\boldsymbol{x}_i)$ generated by the same CNN model and $\boldsymbol{X}_i := \mathscr{T}(\boldsymbol{x}_i)$ the corresponding tensorized input. Suppose $n \geqslant (\sum_{\ell=1}^D d_\ell) \log D$. Then the empirical tensor $\boldsymbol{T}_n$ and population tensor $\boldsymbol{T}$ defined based on this data set obey

$$\left|\!\left|\!\left| \boldsymbol{T}_n - \boldsymbol{T} \right|\!\right|\!\right| := \left|\!\left|\!\left| \frac{1}{n}\sum_{i=1}^n (y_i - y_{\text{avg}})\boldsymbol{X}_i - \mathbb{E}[y\boldsymbol{X}] \right|\!\right|\!\right| \leqslant c \prod_{\ell=1}^D \left\| \boldsymbol{k}^{(\ell)} \right\|_{\ell_2} \frac{\sqrt{\left(\sum_{\ell=1}^D d_\ell\right) \log D} + t}{\sqrt{n}}, \tag{8.1}$$

with probability at least $1 - 5e^{-\min(t^2, t\sqrt{n}, n)}$, where $c > 0$ is an absolute constant.

To prove this theorem, first note that given labels $\{y_i\}_{i=1}^n \sim y$ and their empirical average $y_{\text{avg}} = n^{-1}\sum_{i=1}^n y_i$, we have $\mathbb{E}[y_{\text{avg}}] = \mathbb{E}[y]$. Hence, $y - y_{\text{avg}} = \mathbf{zm}(y - y_{\text{avg}})$ and we can rewrite the empirical tensor as follows:

$$\begin{aligned}
\boldsymbol{T}_n :=& \frac{1}{n}\sum_{i=1}^n (y_i - y_{\text{avg}})\boldsymbol{X}_i \\
=& \frac{1}{n}\sum_{i=1}^n \mathbf{zm}(y_i - y_{\text{avg}})\boldsymbol{X}_i \\
=& \frac{1}{n}\sum_{i=1}^n \mathbf{zm}(y_i)\boldsymbol{X}_i - \mathbf{zm}(y_{\text{avg}})\left(\frac{1}{n}\sum_{i=1}^n \boldsymbol{X}_i\right).
\end{aligned} \tag{8.2}$$

Recall that the population tensor is equal to $\boldsymbol{T} := \mathbb{E}[y_i\boldsymbol{X}_i]$. Furthermore, $\mathbb{E}[\mathbb{E}[y_i]\boldsymbol{X}_i] = \mathbb{E}[y_i]\,\mathbb{E}[\boldsymbol{X}_i] = 0$. Thus, the population tensor can alternatively be written as $\boldsymbol{T} = E[\mathbf{zm}(y_i)\boldsymbol{X}_i] = \frac{1}{n}\sum_{i=1}^n \mathbb{E}[\mathbf{zm}(y_i)\boldsymbol{X}_i]$.

Combining the latter with (8.2), we conclude that

$$\begin{aligned}\boldsymbol{T}_n - \boldsymbol{T} =& \frac{1}{n}\sum_{i=1}^{n}\left(\mathbf{zm}(y_i)\boldsymbol{X}_i - \mathbb{E}[\mathbf{zm}(y_i)\boldsymbol{X}_i]\right) - \mathbf{zm}(y_{\text{avg}})\left(\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{X}_i\right),\\ =& \frac{1}{n}\sum_{i=1}^{n}\mathbf{zm}(\mathbf{zm}(y_i)\boldsymbol{X}_i) - \mathbf{zm}(y_{\text{avg}})\left(\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{X}_i\right),\\ =& \frac{1}{n}\sum_{i=1}^{n}\mathbf{zm}(\mathbf{zm}(y_i)\boldsymbol{X}_i) - \left(\frac{1}{n}\sum_{i=1}^{n}\mathbf{zm}(y_i)\right)\left(\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{X}_i\right).\end{aligned}$$

Now using the triangular inequality for tensor spectral norm we conclude that

$$|||\boldsymbol{T}_n - \boldsymbol{T}||| \leqslant \left|\left|\left|\frac{1}{n}\sum_{i=1}^{n}\mathbf{zm}(\mathbf{zm}(y_i)\boldsymbol{X}_i)\right|\right|\right| + \left|\left|\left|\left(\frac{1}{n}\sum_{i=1}^{n}\mathbf{zm}(y_i)\right)\left(\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{X}_i\right)\right|\right|\right|.$$

We now state two lemmas to bound each of these terms. The proofs of these lemmas are deferred to Sections 8.2.1 and 8.2.2.

LEMMA 8.5 For $i = 1, 2, \ldots, n$ let $\boldsymbol{x}_i \in \mathbb{R}^p$ be i.i.d. random Gaussian vectors distributed as $\mathcal{N}(\mathbf{0}, \boldsymbol{I}_p)$. Also let $\boldsymbol{X}_i \in \mathbb{R}^{\bigotimes_{\ell=1}^{D} d_\ell}$ be the tensorized version of $\boldsymbol{x}_i$ i.e. $\boldsymbol{X}_i = \mathscr{T}(\boldsymbol{x}_i)$. Finally, assume $f : \mathbb{R}^p \mapsto \mathbb{R}$ is an $L$ Lipschitz function. Furthermore, assume $n \geqslant \left(\sum_{\ell=1}^{D} d_\ell\right)\log D$ and $D \geqslant 2$. Then

$$\mathbb{P}\left\{\left|\left|\left|\frac{1}{n}\sum_{i=1}^{n}\mathbf{zm}(\mathbf{zm}(f(\boldsymbol{x}_i))\boldsymbol{X}_i)\right|\right|\right| \leqslant \frac{c_1 L}{\sqrt{n}}\left(\sqrt{\left(\sum_{\ell=1}^{D} d_\ell\right)\log D} + t\right)\right\} \leqslant e^{-\min(t^2, t\sqrt{n})},$$

holds with $c_1 > 0$ a fixed numerical constant.

LEMMA 8.6 Consider the setup of Lemma 8.5. Then

$$\mathbb{P}\left\{\left|\left|\left|\left(\frac{1}{n}\sum_{i=1}^{n}\mathbf{zm}(f(\boldsymbol{x}_i))\right)\left(\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{X}_i\right)\right|\right|\right| \geqslant \frac{c_2 t_1 L}{n}\left(\sqrt{\left(\sum_{\ell=1}^{D} d_\ell\right)\log D} + t_2\right)\right\} \leqslant 2\left(e^{-t_1^2} + e^{-t_2^2}\right)$$

holds with $c_2 > 0$ a fixed numerical constant.

Combining Lemma 8.5 with $f = f_{\text{CNN}}()$, $L = \sum_{r=1}^{R}\prod_{\ell=1}^{D}\left\|\boldsymbol{k}^{(r,\ell)}\right\|_{\ell_2}$ and $c_1 = c/2$ together with Lemma 8.6 with $t_1 = \sqrt{n}$, $t_2 = t$ and $c_2 = c/2$ concludes the proof of Theorem 8.4. All that remains is to prove Lemmas 8.5 and 8.6 which are the subjects of the next two sections.

8.2.1 *Proof of Lemma 8.5* It is more convenient to carry out the steps of the proof on $\sum_{i=1}^{n}\mathbf{zm}(\mathbf{zm}(f(\boldsymbol{x}_i))\boldsymbol{X}_i)$ in lieu of $\frac{1}{n}\sum_{i=1}^{n}\mathbf{zm}(\mathbf{zm}(f(\boldsymbol{x}_i))\boldsymbol{X}_i)$. The lemma trivially follows by a scaling

by a factor $1/n$. We first write the tensor spectral norm as a supremum

$$\left|\left|\left|\sum_{i=1}^{n}\mathbf{zm}(\mathbf{zm}(f(\boldsymbol{x}_i))\boldsymbol{X}_i)\right|\right|\right| = \sup_{\boldsymbol{T}\in\mathcal{RO}}\left|\sum_{i=1}^{n}\left\langle\mathbf{zm}(\mathbf{zm}(f(\boldsymbol{x}_i))\boldsymbol{X}_i),\boldsymbol{T}\right\rangle\right|. \tag{8.3}$$

Let $\boldsymbol{Y}_i = \mathbf{zm}(f(\boldsymbol{x}_i))\boldsymbol{X}_i$. Define the random process $g(\boldsymbol{T}) = \sum_{i=1}^{n}\left\langle\mathbf{zm}(\boldsymbol{Y}_i),\boldsymbol{T}\right\rangle$. We claim that $g(\boldsymbol{T})$ has a mixture of subgaussian and sub-exponential increments (see Definition 8.1 for subgaussian and sub-exponential random variables). Pick two tensors $\boldsymbol{T},\boldsymbol{H}\in\mathbb{R}^{\bigotimes_{\ell=1}^{D}d_\ell}$. Increments of $g$ satisfy the linear relation

$$g(\boldsymbol{T}) - g(\boldsymbol{H}) = \sum_{i=1}^{n}\left\langle\mathbf{zm}(\boldsymbol{Y}_i),\boldsymbol{T}-\boldsymbol{H}\right\rangle.$$

By construction $\mathbb{E}[g(\boldsymbol{T}) - g(\boldsymbol{H})] = 0$. We next claim that $\boldsymbol{Y}_i$ is a sub-exponential vector. Consider a tensor $\boldsymbol{T}$ with unit length $\|\boldsymbol{T}\|_F = 1$ i.e. the sum of squares of entries are equal to one. We have $\left\langle\boldsymbol{Y}_i,\boldsymbol{T}\right\rangle = \mathbf{zm}(f(\boldsymbol{x}_i))\left\langle\boldsymbol{X}_i,\boldsymbol{T}\right\rangle$. $f(\boldsymbol{X}_i)$ is a Lipschitz function of a Gaussian random vector. Thus, by the concentration of Lipschitz functions of Gaussians, we have

$$\mathbb{P}(|\mathbf{zm}(f(\boldsymbol{X}_i))|\geqslant t)\leqslant 2\exp(-\frac{t^2}{2L^2}). \tag{8.4}$$

This immediately implies that $\|\mathbf{zm}(f(\boldsymbol{X}_i))\|_{\psi_2}\leqslant cL$ for a fixed numerical constant $c$. Also note that $\left\langle\boldsymbol{X}_i,\boldsymbol{T}\right\rangle\sim\mathcal{N}(0,1)$ hence $\|\left\langle\boldsymbol{X}_i,\boldsymbol{T}\right\rangle\|_{\psi_2}\leqslant c$. These two identities combined with Lemma 8.3 implies a bound on the sub-exponential norm

$$\|\mathbf{zm}(\left\langle\boldsymbol{Y}_i,\boldsymbol{T}\right\rangle)\|_{\psi_1}\leqslant cL.$$

Next, we observe that $g(\boldsymbol{T})-g(\boldsymbol{H})$ is sum of $n$ i.i.d. sub-exponentials each obeying $\|\left\langle\mathbf{zm}(\boldsymbol{Y}_i),\boldsymbol{T}-\boldsymbol{H}\right\rangle\|_{\psi_1}\leqslant cL\|\boldsymbol{T}-\boldsymbol{H}\|_F$. Applying a standard sub-exponential Bernstein inequality, we conclude that

$$\mathbb{P}(|g(\boldsymbol{T})-g(\boldsymbol{H})|\geqslant t)\leqslant 2\exp\left(-c\cdot\min\left(\frac{t}{L\|\boldsymbol{T}-\boldsymbol{H}\|_F},\frac{t^2}{nL^2\|\boldsymbol{T}-\boldsymbol{H}\|_F^2}\right)\right) \tag{8.5}$$

holds with $\gamma$ a fixed numerical constant. This tail bound implies that $g$ is a mixed tail process that is studied by Talagrand and others [14, 46]. In particular, supremum of such processes is characterized in terms of a linear combination of Talagrand's $\gamma_1$ and $\gamma_2$ functionals (see Definition 8.3 as well as [45, 46] for an exposition). We pick the following distance metrics on tensors induced by the Frobenius norm: $d_1(\boldsymbol{T},\boldsymbol{H}) = L\|\boldsymbol{H}-\boldsymbol{T}\|_F/c$ and $d_2(\boldsymbol{T},\boldsymbol{H}) = \|\boldsymbol{H}-\boldsymbol{T}\|_F L\sqrt{n/c}$. We can thus rewrite (8.5) in the form

$$\mathbb{P}\left\{|g(\boldsymbol{T})-g(\boldsymbol{H})|\geqslant t\right\}\leqslant 2\exp\left(-\min\left(\frac{t}{d_1(\boldsymbol{T},\boldsymbol{H})},\frac{t^2}{d_2^2(\boldsymbol{T},\boldsymbol{H})}\right)\right),$$

which implies $\mathbb{P}\{|g(\boldsymbol{T})-g(\boldsymbol{H})| \geqslant \sqrt{t}d_2(\boldsymbol{T},\boldsymbol{H})+td_1(\boldsymbol{T},\boldsymbol{H})\} \leqslant 2\exp(-t)$. Observe that the radius of $\mathscr{R}\mathscr{O}$ with respect to $\|\cdot\|_F$ norm is 1 hence radius with respect to $d_1, d_2$ metrics are $L/c, L\sqrt{n/c}$, respectively. Applying [14, Theorem 3.5], we obtain

$$\mathbb{P}\left\{ \sup_{\boldsymbol{T}\in\mathscr{R}\mathscr{O}} |g(\boldsymbol{T})| \geqslant C\left(\gamma_2(\mathscr{R}\mathscr{O}, d_2) + \gamma_1(\mathscr{R}\mathscr{O}, d_1) + L\sqrt{un/c} + uL/c\right)\right\} \leqslant e^{-u}.$$

Observe that we can use the change of variable $t = L\cdot\max\left(\sqrt{un}, u\right)$ to obtain

$$\mathbb{P}\left\{ \sup_{\boldsymbol{T}\in\mathscr{R}\mathscr{O}} |g(\boldsymbol{T})| \geqslant C\left(\gamma_2(\mathscr{R}\mathscr{O}, d_2) + \gamma_1(\mathscr{R}\mathscr{O}, d_1) + t\right)\right\} \leqslant \exp\left(-\min\left(\frac{t^2}{L^2 n}, \frac{t}{L}\right)\right), \tag{8.6}$$

with some updated constant $C > 0$. To conclude, we need to bound the $\gamma_2$ and $\gamma_1$ terms. To achieve this, we will upper bound the $\gamma_\alpha$ functional in terms of Dudley's entropy integral which is stated in Lemma 8.4. First, let us find the $\varepsilon$ covering number of $\mathscr{R}\mathscr{O}$. Pick $0 < \delta \leqslant 1$ coverings $\mathscr{C}_\ell$ of the unit $\ell_2$ balls $\mathscr{B}^{d_\ell}$. These covers have size at most $(1+2/\delta)^{d_\ell}$. Consider the set of rank 1 tensors $\mathscr{C} = \mathscr{C}_1 \bigotimes \cdots \bigotimes \mathscr{C}_D$ with size $(1+2/\delta)^{\sum_{\ell=1}^D d_\ell}$. For any $\bigotimes_{\ell=1}^D \boldsymbol{v}_\ell \in \mathscr{R}\mathscr{O}$, we can pick $\bigotimes_{\ell=1}^D \boldsymbol{u}_\ell \in \mathscr{C}$ satisfying $\|\boldsymbol{v}_\ell - \boldsymbol{u}_\ell\|_{\ell_2} \leqslant \delta$ for all $1 \leqslant \ell \leqslant D$. This implies

$$\|(\{\boldsymbol{v}_\ell\}_{\ell=1}^D) - (\{\boldsymbol{u}_\ell\}_{\ell=1}^D)\|_F \leqslant \sum_{\ell=1}^D \|(\boldsymbol{v}_1,\ldots,\boldsymbol{v}_\ell,\boldsymbol{u}_{\ell+1},\ldots,\boldsymbol{u}_D) - (\boldsymbol{v}_1,\ldots,\boldsymbol{v}_{\ell-1},\boldsymbol{u}_\ell,\ldots,\boldsymbol{u}_D)\|_F \tag{8.7}$$

$$= \sum_{\ell=1}^D \|\boldsymbol{v}_\ell\|_{\ell_2}\ldots\|\boldsymbol{v}_{\ell-1}\|_{\ell_2}\|\boldsymbol{v}_\ell - \boldsymbol{u}_\ell\|_{\ell_2}\|\boldsymbol{u}_{\ell+1}\|_{\ell_2}\ldots\|\boldsymbol{u}_D\|_{\ell_2} \leqslant D\delta. \tag{8.8}$$

Denoting Frobenius norm covering number of $\mathscr{R}\mathscr{O}$ by $N(\varepsilon)$, this implies that, for $0 < \varepsilon \leqslant 1$,

$$N(\varepsilon) \leqslant (1+2D/\varepsilon)^{\sum_{\ell=1}^D d_\ell}.$$

Clearly, $N(\varepsilon) = 1$ for $\varepsilon \geqslant 1$ by picking the cover $\{0\}$. Consequently,

$$\gamma_1(\mathscr{R}\mathscr{O}, \|\cdot\|_F) \leqslant \int_0^1 \log N(\varepsilon)\,\mathrm{d}\varepsilon \leqslant c\left(\sum_{\ell=1}^D d_\ell\right)\log D,\ \gamma_2(\mathscr{R}\mathscr{O}, \|\cdot\|_F) \leqslant \int_0^1 \sqrt{\log N(\varepsilon)}\,\mathrm{d}\varepsilon$$

$$\leqslant c\sqrt{\left(\sum_{\ell=1}^D d_\ell\right)\log D}. \tag{8.9}$$

Thus, the metrics $d_1, d_2$ metrics are $\|\cdot\|_F$ norm scaled by a constant. Hence, their $\gamma_\alpha$ functions are scaled versions of (8.9) given by

$$\gamma_1(\mathscr{R}\mathscr{O}, d_1) \leqslant cL\left(\sum_{\ell=1}^D d_\ell\right)\log D,\ \gamma_2(\mathscr{R}\mathscr{O}, d_2) \leqslant cL\sqrt{n\left(\sum_{\ell=1}^D d_\ell\right)\log D}.$$

Now, observe that if $n \geqslant \left(\sum_{\ell=1}^{D} d_\ell\right) \log D$, we have $\gamma_1(\mathscr{R}\mathscr{O}, d_1) \leqslant cL\sqrt{n\left(\sum_{\ell=1}^{D} d_\ell\right) \log D}$. Substituting these in (8.6) and using $n \geqslant \left(\sum_{\ell=1}^{D} d_\ell\right) \log D$, we find

$$\mathbb{P}\left\{ \sup_{\boldsymbol{T} \in \mathscr{R}\mathscr{O}} |g(\boldsymbol{T})| \geqslant C\left(L\sqrt{n\left(\sum_{\ell=1}^{D} d_\ell\right) \log D} + t\right) \right\} \leqslant e^{-\min\left(\frac{t^2}{L^2 n}, \frac{t}{L}\right)}.$$

Substituting $t \to L\sqrt{n}t$ and recalling (8.3) conclude the proof.

8.2.2 *Proof of Lemma 8.6* We first rewrite

$$\left\| \left(\frac{1}{n}\sum_{i=1}^{n} \mathbf{zm}(f(\boldsymbol{x}_i))\right)\left(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{X}_i\right) \right\| = \left| \frac{1}{n}\sum_{i=1}^{n} \mathbf{zm}(f(\boldsymbol{x}_i)) \right| \cdot \left\| \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{X}_i \right\|. \tag{8.10}$$

As discussed in (8.4), $\|\mathbf{zm}(f(\boldsymbol{x}_i))\|_{\psi_2} \leqslant cL$ for $c$ a fixed numerical constant. Since $\boldsymbol{x}_i$s are i.i.d., the empirical sum $\boldsymbol{f}_{\text{avg}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{zm}(f(\boldsymbol{x}_i))$ obeys the bound $\|\boldsymbol{f}_{\text{avg}}\|_{\psi_2} \leqslant cL/\sqrt{n}$ as well. Hence,

$$\mathbb{P}\left\{ \left|\boldsymbol{f}_{\text{avg}}\right| \geqslant c' \frac{t_1 L}{\sqrt{n}} \right\} \leqslant 2e^{-t_1^2}. \tag{8.11}$$

Also note that $\frac{1}{\sqrt{n}}\sum_{i=1}^{n} \boldsymbol{X}_i$ is a tensor with standard normal entries; applying [47, Theorem 1], we conclude that

$$\left\| \frac{1}{\sqrt{n}}\sum_{i=1}^{n} \boldsymbol{X}_i \right\| \leqslant c''\left(\sqrt{\left(\sum_{\ell=1}^{D} d_\ell\right) \log D} + t_2\right) \quad \Rightarrow \quad \left\| \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{X}_i \right\| \leqslant c'' \frac{\sqrt{\left(\sum_{\ell=1}^{D} d_\ell\right) \log D} + t_2}{\sqrt{n}} \tag{8.12}$$

holds with probability $1 - 2e^{-t_2^2}$. Combining (8.11) and (8.12) via the union bound together with (8.10) concludes the proof.

8.3 *Rank-R approximation of the population tensor (Proof of Theorem 4.7)*

We will prove the following more general version of this theorem that allows for arbitrary kernel norms.

THEOREM 8.5 Consider the setup of Theorem 8.4. Also, assume the activations are $S$-smooth per Definition 4.5 and the convolutional kernels are $\mu$-diffused per Definition 4.6. Then, the population tensor $\boldsymbol{T} := \mathbb{E}[y\boldsymbol{X}]$ can be approximated by the rank-$R$ tensor $\boldsymbol{L}_{\text{CNN}} := \sum_{r=1}^{R} \alpha_{r,\text{CNN}} \bigotimes_{\ell=1}^{D} \boldsymbol{k}^{(r,\ell)}$ as follows:

$$\|\boldsymbol{T} - \boldsymbol{L}_{\text{CNN}}\|_F \leqslant \sqrt{8\pi}\mu S \cdot \sum_{r=1}^{R}\prod_{\ell=1}^{D} \left\|\boldsymbol{k}^{(r,\ell)}\right\|_{\ell_2} \cdot \sup_{r,i} \prod_{i=1}^{\ell} \left\|\boldsymbol{k}^{(r,\ell)}\right\|_{\ell_2} \frac{D}{\sqrt{\min_{\ell} d_\ell}}. \tag{8.13}$$

We first prove the result by reduction to the single-filter CNN model. First, we have the following result.

THEOREM 8.6 Consider the setup of Theorem 4.2 with the single-filter model ($R = 1$) and convolutional filters $(\boldsymbol{k}^{(\ell)})_{\ell=1}^{D}$. Also, assume the activations are $S$-smooth per Definition 4.5 and the convolutional kernels are $\mu$-diffused per Definition 4.6. Then, the population tensor $\boldsymbol{T} := \mathbb{E}[y\boldsymbol{X}]$ can be approximated by the rank-1 tensor $\boldsymbol{L}_{\text{CNN}} := \alpha_{\text{CNN}} \bigotimes_{\ell=1}^{D} \boldsymbol{k}^{(\ell)}$ as follows:

$$\left|\!\left|\!\left| \boldsymbol{T} - \boldsymbol{L}_{\text{CNN}} \right|\!\right|\!\right| \leqslant \|\boldsymbol{T} - \boldsymbol{L}_{\text{CNN}}\|_F \leqslant \sqrt{8\pi}\mu S \cdot \prod_{\ell=1}^{D} \left\|\boldsymbol{k}^{(\ell)}\right\|_{\ell_2} \cdot \sup_i \prod_{i=1}^{\ell} \left\|\boldsymbol{k}^{(\ell)}\right\|_{\ell_2} \frac{D}{\sqrt{\min_\ell d_\ell}}.$$

Based on this theorem, we can prove the main result as follows:

*Proof of Theorem 8.5*: we will utilize the fact that $R$-filter CNN is a linear combination of $R$ single-filter CNN as described in (2.2). From the linearity of expectation, □

$$\boldsymbol{T} = \sum_{r=1}^{R} \boldsymbol{T}_r \quad \text{where} \quad \boldsymbol{T}_r = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I}_p)}[f_{SF,r}(\boldsymbol{x})\mathscr{T}(\boldsymbol{x})].$$

Next, denote the rank-1 tensor associated with $r$th single-filter sub-network by

$$\boldsymbol{L}_{\text{CNN,r}} = \alpha_{r,\text{CNN}} \bigotimes_{\ell=1}^{D} \boldsymbol{k}^{(r,\ell)}.$$

Theorem 8.6 guarantees that, for each $1 \leqslant r \leqslant R$,

$$\left|\!\left|\!\left| \boldsymbol{T}_r - \boldsymbol{L}_{\text{CNN,r}} \right|\!\right|\!\right| \leqslant \|\boldsymbol{T}_r - \boldsymbol{L}_{\text{CNN,r}}\|_F \leqslant \sqrt{8\pi}\mu S \cdot \prod_{\ell=1}^{D} \left\|\boldsymbol{k}^{(r,\ell)}\right\|_{\ell_2} \cdot \sup_i \prod_{i=1}^{\ell} \left\|\boldsymbol{k}^{(r,\ell)}\right\|_{\ell_2} \frac{D}{\sqrt{\min_\ell d_\ell}}.$$

Adding the above relations, noticing $\sum_{r=1}^{R} \boldsymbol{L}_{\text{CNN,r}} = \boldsymbol{L}_{\text{CNN}}$ and $\sum_{r=1}^{R} \boldsymbol{T}_r = \boldsymbol{T}$ and using triangle inequality yield the desired bound (8.13).

With this proof out of way, we can focus our attention to proving our single-filter result Theorem 8.6.

*Proof of Theorem 8.6*: we begin the proof of this theorem by a few definitions regarding non-overlapping CNN models that simplify our exposition. For these definitions, it is convenient to view non-overlapping CNNs as a tree with the root of the tree corresponding to the output of the CNN and the leaves corresponding to input features. In this visualization, $D - \ell$th layer of the tree corresponds to the $\ell$th layer. Figure 5 depicts such a tree visualization along with the definitions discussed below. □

DEFINITION 8.7 (Path vector). A vector $\mathbf{i} \in \mathbb{R}^{D+1}$ is a path vector if its zeroth coordinate satisfies $1 \leqslant \mathbf{i}_0 \leqslant p$ and for all $D - 1 \geqslant j \geqslant 0$, $\mathbf{i}_{j+1}$ obeys $\mathbf{i}_{j+1} = \left\lceil \mathbf{i}_j / d_j \right\rceil$. This implies $1 \leqslant \mathbf{i}_j \leqslant p_j$ and $\mathbf{i}_D = 1$. We note that in the tree visualization a path vector corresponds to a path connecting a leaf (input feature) to the root of the tree (output). We use $\mathscr{I}$ to denote the set of all path vectors and note that

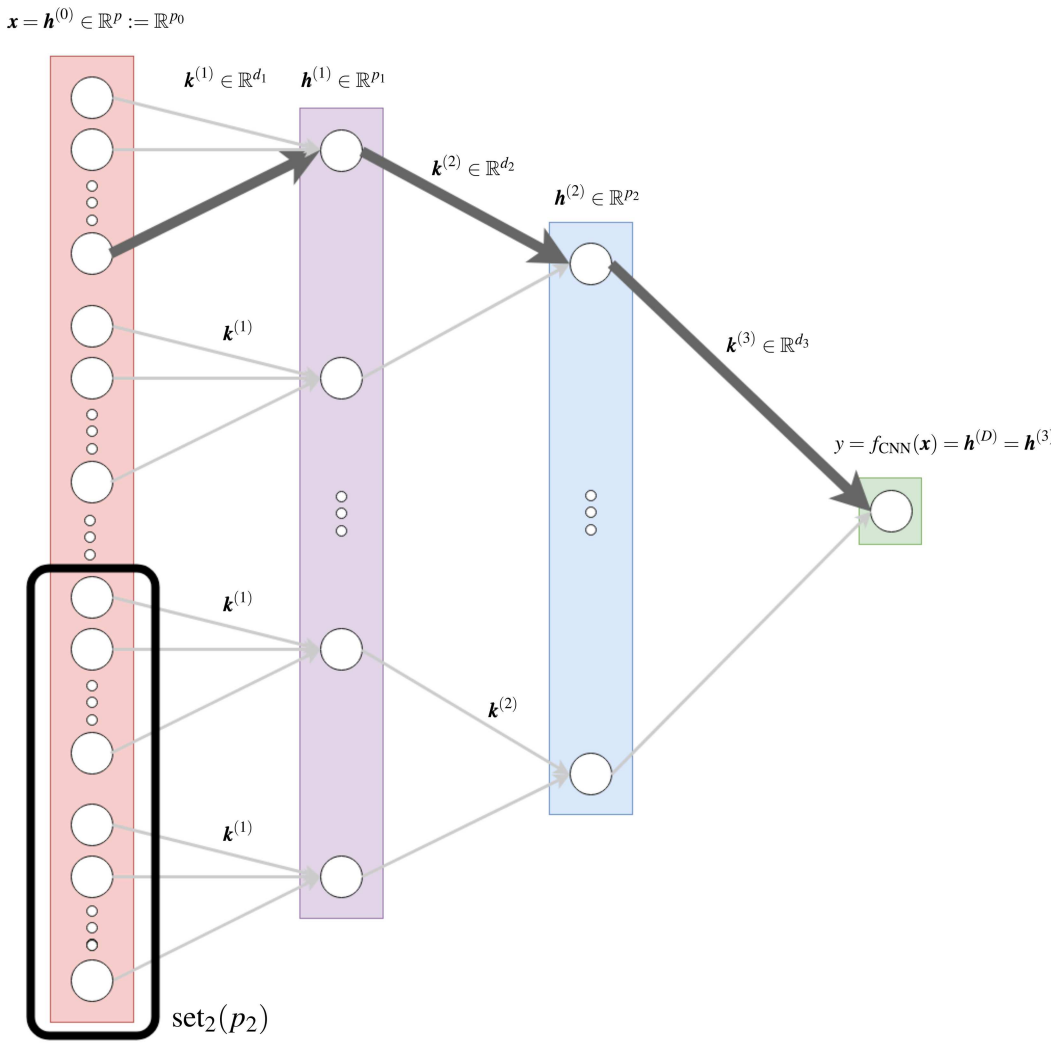



FIG. 5. Tree visualization of a non-overlapping CNN model. The path in bold corresponds to the path vector $\mathbf{i} = (d_1, 1, 1, 1)$ from Definition 8.7. For this path, the kernel path gain is equal $k_{\mathbf{i}} = k_{d_1}^{(1)} k_1^{(2)} k_1^{(3)}$ and the activation path gain is equal to $\phi'_{\mathbf{i}}(x) = \phi'_1(\bar{h}_1^{(1)})\phi'_2(\bar{h}_1^{(2)})\phi'_3(\bar{h}_1^{(3)})$. The $\mathrm{set}_2(p_2)$ (offsprings of the last hidden node in layer two) is outlined.

$|\mathscr{I}| = p$. We also define $\mathbf{i}(i) \in \mathscr{I}$ be the vector whose zeroth entry is $\mathbf{i}_0 = i$. Stated differently $\mathbf{i}(i)$ is the path connecting the input feature $i$ to the output. Given a path $\mathbf{i}$ and a $p$ dimensional vector $\boldsymbol{v}$, we define $\boldsymbol{v}_{\mathbf{i}} := \boldsymbol{v}_{\mathbf{i}_0}$. A sample path vector is depicted in bold in Fig. 5, which corresponds to $\mathbf{i} = (d_1, 1, 1, 1)$.

DEFINITION 8.8 (Kernel and activation path gains). Consider a CNN model of the form (2.2) with input $\boldsymbol{x}$ and inputs of hidden units given by $\{\bar{\boldsymbol{h}}^{(\ell)}\}_{\ell=1}^{D}$. To any path vector $\boldsymbol{i}$ connecting an input feature to the

output, we associate two path gains: a kernel path gain denoted by $\boldsymbol{k}_{\mathbf{i}}$ and activation path gain denoted by $\phi'_{\mathbf{i}}(\boldsymbol{x})$ defined as

$$\boldsymbol{k}_{\mathbf{i}} = \prod_{\ell=1}^{D} \boldsymbol{k}^{(\ell)}_{\mathrm{mod}(\mathbf{i}_{\ell-1},d_\ell)} \quad \text{and} \quad \phi'_{\mathbf{i}}(\boldsymbol{x}) = \prod_{\ell=1}^{D} \phi'_\ell(\bar{\boldsymbol{h}}^{(\ell)}_{\mathbf{i}_\ell}),$$

where $\mathrm{mod}(a,b)$ denotes the remainder of dividing integer $a$ by $b$. In words, the kernel path gain is multiplication of the kernel weights along the path and the activation path gain is the multiplication of the derivatives of the activations evaluated at the hidden units along the path. For the path depicted in Fig. 5 in bold, the kernel path gain is equal $k_{\mathbf{i}} = \boldsymbol{k}^{(1)}_{d_1}\boldsymbol{k}^{(2)}_1\boldsymbol{k}^{(3)}_1$ and the activation path gain is equal to $\phi'_{\mathbf{i}}(\boldsymbol{x}) = \phi'_1(\bar{h}^{(1)}_1)\phi'_2(\bar{h}^{(2)}_1)\phi'_3(\bar{h}^{(3)}_1)$.

DEFINITION 8.9 (CNN offsprings). Consider a CNN model of the form (2.2) with input $\boldsymbol{x}$ and inputs of hidden units given by $\{\bar{\boldsymbol{h}}^{(\ell)}\}_{\ell=1}^{D}$. We will associate $\mathrm{set}_\ell(i) \subset \{1,\dots,p\}$ to the $i$th hidden unit of layer $\ell$ defined as $\mathrm{set}_\ell(i) = \{(i-1)r_\ell + 1, (i-1)r_\ell + 2, \dots, ir_\ell\}$ where $r_\ell = p/p_\ell$. By construction, this corresponds to the set of entries of the input data $\boldsymbol{x}$ that $\bar{\boldsymbol{h}}^{(\ell)}_i(\boldsymbol{x})$ is dependent on. In the tree analogy, $\mathrm{set}_\ell(i)$ are the leaves of the tree connected to hidden unit $i$ in the $\ell$th layer i.e. the set of offsprings of this hidden node. We depict $\mathrm{set}_2(p_2)$ which are the offsprings of the last hidden node in layer two in Fig. 5.

We now will rewrite the population tensor in a form such that it is easier to see why it can be well approximated by a rank one tensor. Note that since the tensorization operation is linear, the population tensor is equal to

$$\boldsymbol{T} = \mathbb{E}[f_{\mathrm{CNN}}(\boldsymbol{x})\boldsymbol{X}] = \mathbb{E}[f_{\mathrm{CNN}}(\boldsymbol{x})\mathscr{T}(\boldsymbol{x})] = \mathscr{T}\left(\mathbb{E}[f_{\mathrm{CNN}}(\boldsymbol{x})\boldsymbol{x}]\right). \tag{8.14}$$

Define the vector $\boldsymbol{g}^{C}NN$ to be the population gradient vector i.e. $\boldsymbol{g}^{C}NN = \mathbb{E}[\nabla f_{\mathrm{CNN}}(\boldsymbol{x})]$ and note that Stein's lemma combined with (8.14) implies that

$$\boldsymbol{T} = \mathscr{T}\left(\mathbb{E}[f_{\mathrm{CNN}}(\boldsymbol{x})\boldsymbol{x}]\right) = \mathscr{T}\left(\mathbb{E}[\nabla f_{\mathrm{CNN}}(\boldsymbol{x})]\right) = \mathscr{T}\left(\boldsymbol{g}^{C}NN\right). \tag{8.15}$$

Also note that

$$\frac{\partial f_{\mathrm{CNN}}(\boldsymbol{x})}{\partial \boldsymbol{x}_i} = \boldsymbol{k}_{\mathbf{i}(i)}\phi'_{\mathbf{i}(i)}(\boldsymbol{x}).$$

Thus, we have

$$\boldsymbol{g}^{C}NN_i = \mathbb{E}\left[\frac{\partial f_{\mathrm{CNN}}(\boldsymbol{x})}{\partial \boldsymbol{x}_i}\right] = \boldsymbol{k}_{\mathbf{i}(i)}\,\mathbb{E}[\phi'_{\mathbf{i}(i)}(\boldsymbol{x})]. \tag{8.16}$$

Define a vector $\boldsymbol{k} \in \mathbb{R}^p$ such that $\boldsymbol{k}_i = \boldsymbol{k}_{\mathbf{i}(i)}$. Since $\boldsymbol{k}_{\mathbf{i}(i)}$ consists of the product of the kernel values across the path $\mathbf{i}(i)$, it is easy to see that the tensor $\boldsymbol{K} := \mathscr{T}(\boldsymbol{k})$ is equal to

$$\boldsymbol{K} = \bigotimes_{\ell=1}^{D} \boldsymbol{k}^{(\ell)}. \tag{8.17}$$

Similarly define the expected path gain vector $\boldsymbol{v} \in \mathbb{R}^p$ such that $\boldsymbol{v}_i = E[\phi'_{\mathbf{i}(i)}(\boldsymbol{x})]$ and define the corresponding expected path gain tensor $\boldsymbol{V} = \mathscr{T}(\boldsymbol{v})$. Here, $\boldsymbol{v}$ and $\boldsymbol{V}$ capture the affect of the nonlinear activations within the population gradient vector. Therefore, (8.16) can be rewritten in the vector form $\boldsymbol{g}^{CNN} = \boldsymbol{k} \odot \boldsymbol{v}$ where $\boldsymbol{a} \odot \boldsymbol{b}$ denotes entry-wise (Hadamard) product between two vectors/matrices/tensors $\boldsymbol{a}$ and $\boldsymbol{b}$ of the same size. Thus, using (8.15), the population tensor can alternatively be written as

$$\boldsymbol{T} = \mathscr{T}\left(\boldsymbol{g}^{CNN}\right) = \boldsymbol{K} \odot \boldsymbol{V} = \left(\bigotimes_{\ell=1}^{D} \boldsymbol{k}^{(\ell)}\right) \odot \boldsymbol{V}.$$

Therefore, the population tensor $\boldsymbol{T}$ is the outer product of the convolutional kernels whose entries are masked with the expected path gain $\boldsymbol{V}$. If the entries of the tensor $\boldsymbol{V}$ were all the same, the population tensor would be exactly rank one with the factors revealing the convolutional kernel. One natural choice for approximating the population tensor with a rank one matrix is to replace the masking tensor $\boldsymbol{V}$ with a scalar. That is, use the approximation $\boldsymbol{T} \approx c \bigotimes_{\ell=1}^{D} \boldsymbol{k}^{(\ell)}$. Recall that $\boldsymbol{L}_{\text{CNN}} := \alpha_{\text{CNN}} \bigotimes_{\ell=1}^{D} \boldsymbol{k}^{(\ell)}$ is exactly such an approximation with $c$ set to $\alpha_{\text{CNN}}$ given by

$$\alpha_{\text{CNN}} = \prod_{\ell=1}^{D} \mathbb{E}_{\boldsymbol{x} \sim \mathscr{N}(0,\boldsymbol{I})}[\phi'_\ell(\bar{\boldsymbol{h}}_{\mathbf{i}_\ell}^{(\ell)})].$$

To characterize the quality of this approximation, note that

$$\begin{aligned}
\left|\left|\left|\boldsymbol{T} - \boldsymbol{L}_{\text{CNN}}\right|\right|\right| &\overset{(a)}{\leqslant} \left\|\boldsymbol{T} - \boldsymbol{L}_{\text{CNN}}\right\|_F, \\
&\overset{(b)}{=} \left\|\boldsymbol{K} \odot \left(\boldsymbol{V} - \alpha_{\text{CNN}}\right)\right\|_F, \\
&\overset{(c)}{=} \left\|\boldsymbol{k} \odot \left(\boldsymbol{v} - \alpha_{\text{CNN}}\right)\right\|_{\ell_2}, \\
&\overset{(d)}{\leqslant} \|\boldsymbol{k}\|_{\ell_2} \left\|\boldsymbol{v} - \alpha_{\text{CNN}}\right\|_{\ell_\infty}, \\
&\overset{(e)}{=} \prod_{\ell=1}^{D} \left\|\boldsymbol{k}^{(\ell)}\right\|_{\ell_2} \left\|\boldsymbol{v} - \alpha_{\text{CNN}}\right\|_{\ell_\infty}.
\end{aligned}$$

Here, (a) follows from the fact for a tensor, its spectral norm is smaller than its Frobenius norm, (b) from the definitions of $\boldsymbol{T}$ and $\boldsymbol{L}_{\text{CNN}}$, (c) from the fact that $\boldsymbol{K} = \mathscr{T}(\boldsymbol{k})$ and $\boldsymbol{V} = \mathscr{T}(\boldsymbol{v})$, (d) from the fact that $\|\boldsymbol{a} \odot \boldsymbol{v}\|_{\ell_2} \leqslant \|\boldsymbol{a}\|_{\ell_2} \|\boldsymbol{b}\|_{\ell_\infty}$ and (e) from the fact that the Euclidean norm of the Kronecker product of vectors is equal to the product of the Euclidean norm of the individual vectors. As a result of the latter inequality to prove Theorem 4.7, it suffices to show that

$$\left\|\boldsymbol{v} - \alpha_{\text{CNN}}\right\|_{\ell_\infty} \leqslant \sqrt{8\pi}\mu S \cdot \sup_{\ell} \prod_{i=1}^{\ell} \left\|\boldsymbol{k}^{(i)}\right\|_{\ell_2} \cdot \frac{D}{\sqrt{\min_{\ell} d_\ell}}. \tag{8.18}$$

In the next lemma, we prove a stronger statement.

LEMMA 8.7 Assume the activations are $S$-smooth. Also consider a vector $\boldsymbol{v} \in \mathbb{R}^p$ with entries $\boldsymbol{v}_i = E_{\boldsymbol{x}\sim\mathcal{N}(0,\boldsymbol{I})}[\phi'_{\mathbf{i}(i)}(\boldsymbol{x})]$ and $\alpha_{\mathrm{CNN}} = \prod_{\ell=1}^{D} \mathbb{E}_{\boldsymbol{x}\sim\mathcal{N}(0,\boldsymbol{I})}[\phi'_\ell(\bar{\boldsymbol{h}}^{(\ell)}_{\mathbf{i}_\ell})]$ we have

$$\left|\boldsymbol{v}_i - \alpha_{\mathrm{CNN}}\right| \leqslant \kappa_i := \sqrt{8\pi} S \sum_{\ell=1}^{D} \left|\boldsymbol{k}^{(\ell)}_{\mathrm{mod}(\mathbf{i}_{\ell-1},d_\ell)}\right| \prod_{i=1}^{\ell-1} \|\boldsymbol{k}^{(i)}\|_{\ell_2}.$$

Here, $\boldsymbol{i}$ is the vector path that starts at input feature $i$.

Before proving this lemma, let us explain how (8.18) follows from this lemma. To show this, we use the kernel diffuseness assumption introduced in Definition 4.6. This definition implies that $\left|\boldsymbol{k}^{(\ell)}_{\mathrm{mod}(\mathbf{i}_{\ell-1},d_\ell)}\right| \leqslant \left\|\boldsymbol{k}^{(\ell)}\right\|_{\ell_\infty} \leqslant \frac{\mu}{\sqrt{d_\ell}} \left\|\boldsymbol{k}^{(\ell)}\right\|_{\ell_2}$. Thus, we have

$$\begin{aligned}
\kappa_i :=& \sqrt{8\pi} S \sum_{\ell=1}^{D} |\boldsymbol{k}^{(\ell)}_{\mathrm{mod}(\mathbf{i}_{\ell-1},d_\ell)}| \prod_{i=1}^{\ell-1} \|\boldsymbol{k}^{(i)}\|_{\ell_2}, \\
\leqslant& \sqrt{8\pi}\mu S \sum_{\ell=1}^{D} \frac{1}{\sqrt{d_\ell}} \left\|\boldsymbol{k}^{(\ell)}\right\|_{\ell_2} \prod_{i=1}^{\ell-1} \|\boldsymbol{k}^{(i)}\|_{\ell_2}, \\
=& \sqrt{8\pi}\mu S \sum_{\ell=1}^{D} \frac{1}{\sqrt{d_\ell}} \prod_{i=1}^{\ell} \|\boldsymbol{k}^{(i)}\|_{\ell_2}, \\
\leqslant& \sqrt{8\pi}\mu D S \cdot \sup_\ell \frac{\prod_{i=1}^{\ell} \|\boldsymbol{k}^{(i)}\|_{\ell_2}}{\sqrt{d_\ell}}, \\
\leqslant& \sqrt{8\pi}\mu S \cdot \sup_\ell \prod_{i=1}^{\ell} \left\|\boldsymbol{k}^{(i)}\right\|_{\ell_2} \cdot \frac{D}{\sqrt{\min_\ell d_\ell}}.
\end{aligned}$$

This completes the proof of Theorem 4.7. All that remains is to prove Lemma 8.7 which is the subject of the next section.

8.3.1 *Proof of Lemma 8.7* To bound the difference between $\boldsymbol{v}_i$ and $\alpha_{\mathrm{CNN}}$, consider the path $\mathbf{i} = \mathbf{i}(i)$ and define the variables $\{a_i\}_{i=0}^{D}$ as

$$a_i = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{N}(0,\boldsymbol{I})}[\prod_{\ell=1}^{i} \phi'_\ell(\bar{\boldsymbol{h}}^{(\ell)}_{\mathbf{i}_\ell})] \prod_{\ell=i+1}^{D} \mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\boldsymbol{I})}[\phi'_\ell(\bar{\boldsymbol{h}}^{(\ell)}_{\mathbf{i}_\ell})].$$

Note that $a_D = \boldsymbol{v}_i$ and $a_0 = \alpha_{\mathrm{CNN}}$. To bound the difference $\boldsymbol{v}_i - \alpha_{\mathrm{CNN}} = a_D - a_0$, we use a telescopic sum

$$|a_D - a_0| \leqslant \sum_{\ell=0}^{D-1} |a_{\ell+1} - a_\ell|. \tag{8.19}$$

We thus focus on bounding each of the summands $|a_\ell - a_{\ell-1}|$. Setting $\gamma_\ell = \prod_{i=\ell}^{D} \mathbb{E}_{\boldsymbol{x}\sim\mathcal{N}(0,\boldsymbol{I})}[\phi_i'(\bar{\boldsymbol{h}}_{\mathbf{i}_i}^{(i)})]$, this can be written as

$$a_\ell - a_{\ell-1} = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{N}(0,\boldsymbol{I})}[(\phi_\ell'(\bar{\boldsymbol{h}}_{\mathbf{i}_\ell}^{(\ell)}) - \mathbb{E}[\phi_\ell'(\bar{\boldsymbol{h}}_{\mathbf{i}_\ell}^{(\ell)})]) \prod_{i=1}^{\ell-1} \phi_\ell'(\bar{\boldsymbol{h}}_{\mathbf{i}_\ell}^{(\ell)})]\gamma_{\ell+1}.$$

Using $|\gamma_\ell| \leqslant 1$ (which follows from the assumption that the activations are 1-Lipschitz), it suffices to bound

$$\mathbb{E}_{\boldsymbol{x}\sim\mathcal{N}(0,\boldsymbol{I})}[(\phi_\ell'(\bar{\boldsymbol{h}}_{\mathbf{i}_\ell}^{(\ell)}) - \mathbb{E}[\phi_\ell'(\bar{\boldsymbol{h}}_{\mathbf{i}_\ell}^{(\ell)})])\theta_{\ell-1}] \quad \text{where} \quad \theta_\ell = \prod_{i=1}^{\ell} \phi_i'(\bar{\boldsymbol{h}}_{\mathbf{i}_i}^{(i)}). \tag{8.20}$$

To this aim, we state two useful lemmas whose proofs are deferred to Sections 8.3.1 and 8.3.1.

LEMMA 8.8 Let $X, Y, Z$ be random variables where $X$ is independent of $Z$. Let $f$ be an $L$-Lipschitz function. Then

$$|\,\mathbb{E}[f(X+Y)Z] - \mathbb{E}[f(X+Y)]\,\mathbb{E}[Z]| \leqslant L\,\mathbb{E}[|\mathbf{zm}(Y)|(|Z| + |\,\mathbb{E}[Z]|)]. \tag{8.21}$$

Furthermore, if $|Z| \leqslant 1$, then

$$|\,\mathbb{E}[f(X+Y)Z] - \mathbb{E}[f(X+Y)]\,\mathbb{E}[Z]| \leqslant 2L\,\mathbb{E}[|\mathbf{zm}(Y)|]. \tag{8.22}$$

LEMMA 8.9 $\bar{\boldsymbol{h}}_i^{(\ell)}(\boldsymbol{x})$ (and $\boldsymbol{h}_i^{(\ell)}(\boldsymbol{x})$) is a deterministic function of the entries of $\boldsymbol{x}$ indexed by $\mathrm{set}_\ell(i)$. In other words, there exists a function $f$ such that $\bar{\boldsymbol{h}}_i^{(\ell)}(\boldsymbol{x}) = f(\boldsymbol{x}_{\mathrm{set}_\ell(i)})$.

With these lemmas in hand, we return to bounding (8.20). To this aim, we decompose $\bar{\boldsymbol{h}}_{\mathbf{i}_\ell}^{(\ell)}$ as follows:

$$\bar{\boldsymbol{h}}_{\mathbf{i}_\ell}^{(\ell)} = \sum_{i=1}^{d_\ell} \boldsymbol{k}_i^{(\ell)} \boldsymbol{h}_{d_\ell(\mathbf{i}_\ell-1)+i}^{(\ell-1)} = \boldsymbol{k}_{\mathrm{mod}(\mathbf{i}_{\ell-1},d_\ell)}^{(\ell)} \boldsymbol{h}_{\mathbf{i}_{\ell-1}}^{(\ell-1)} + \mathbf{r},$$

where the $\mathbf{r}$ term is the contribution of the entries of $\boldsymbol{h}^{(\ell-1)}$ other than $\mathbf{i}_{\ell-1}$. By the non-overlapping assumption, $\mathbf{r}$ is independent of $\theta_{\ell-1}$ as well as $\boldsymbol{h}_{\mathbf{i}_{\ell-1}}^{(\ell-1)}$ (see Lemma 8.9). In particular, $\boldsymbol{h}_{\mathbf{i}_{\ell-1}}^{(\ell-1)}$ and $\theta_{\ell-1}$ are functions of $\boldsymbol{x}_{\mathrm{set}_{\ell-1}(\mathbf{i}_{\ell-1})}$ whereas $\mathbf{r}$ is a function of the entries over the complement $\mathrm{set}_\ell(\mathbf{i}_\ell) - \mathrm{set}_{\ell-1}(\mathbf{i}_{\ell-1})$. With these observations, applying Lemma 8.8 with $f = \phi_\ell', X = \mathbf{r}, Y = \boldsymbol{k}_{\mathrm{mod}(\mathbf{i}_{\ell-1},d_\ell)}^{(\ell)} \boldsymbol{h}_{\mathbf{i}_{\ell-1}}^{(\ell-1)}, Z = \theta_{\ell-1}$ and using the fact that $|\theta_{\ell-1}| \leqslant 1$ which holds due to 1-Lipschitzness of $\sigma_\ell$'s, we conclude that

$$|\,\mathbb{E}_{\boldsymbol{x}\sim\mathcal{N}(0,\boldsymbol{I})}[(\phi_\ell'(\bar{\boldsymbol{h}}_{\mathbf{i}_\ell}^{(\ell)}) - \mathbb{E}[\phi_\ell'(\bar{\boldsymbol{h}}_{\mathbf{i}_\ell}^{(\ell)})])\theta_{\ell-1}]| \leqslant 2S\,\mathbb{E}\,|\mathbf{zm}(Y)|.$$

Here, $S$ is the smoothness of $\sigma_\ell$ and Lipschitz constant of $\phi_\ell'$. To conclude, we need to assess the $\mathbb{E}\,|\mathbf{zm}(Y)|$ term. Now note that starting from $\boldsymbol{x}$, each entry of $\boldsymbol{h}^{(\ell-1)}$ is obtained by applying a sequence of inner products with $\{\boldsymbol{k}^{(i)}\}_{i=1}^{\ell-1}$ and activations $\sigma_\ell(\cdot)$, which implies $\boldsymbol{h}_{\mathbf{i}_{\ell-1}}^{(\ell-1)}$ is a $\prod_{i=1}^{\ell-1} \|\boldsymbol{k}^{(i)}\|_{\ell_2}$-Lipschitz

function of $\text{set}_{\ell-1}(\mathbf{i}_{\ell-1})$. This implies $Y$ is a Lipschitz function of a Gaussian vector with Lipschitz constant $L_Y = |\boldsymbol{k}^{(\ell)}_{\text{mod}(\mathbf{i}_{\ell-1},d_\ell)}| \prod_{i=1}^{\ell-1} \|\boldsymbol{k}^{(i)}\|_{\ell_2}$. Hence, $\mathbf{zm}(Y)$ obeys the tail bound

$$\mathbb{P}(|\mathbf{zm}(Y)| \geqslant t) \leqslant 2\exp(-\frac{t^2}{2L_Y^2}).$$

Using a standard integration by parts argument the latter implies that

$$\mathbb{E}\,|\mathbf{zm}(Y)| \leqslant \sqrt{2\pi} L_Y.$$

Thus,

$$\left|a_\ell - a_{\ell-1}\right| \leqslant |\boldsymbol{k}^{(\ell)}_{\text{mod}(\mathbf{i}_{\ell-1},d_\ell)}| \prod_{i=1}^{\ell-1} \|\boldsymbol{k}^{(i)}\|_{\ell_2},$$

concluding the upper bound on each summand of (8.19). Combining such upper bounds (8.19) implies

$$|a_D - a_0| \leqslant \sqrt{8\pi} S \sum_{\ell=1}^{D} |\boldsymbol{k}^{(\ell)}_{\text{mod}(\mathbf{i}_{\ell-1},d_\ell)}| \prod_{i=1}^{\ell-1} \|\boldsymbol{k}^{(i)}\|_{\ell_2} := \kappa_{\mathbf{i}}.$$

This concludes the proof of Lemma 8.7.

8.3.1.1. Proof of Lemma 8.8 Using the independence of $X, Z$, we can write

$$\begin{aligned}\mathbb{E}[f(X+Y)Z] &= \mathbb{E}[f(X+\mathbb{E}[Y])Z] + \mathbb{E}[(f(X+Y) - f(X+\mathbb{E}[Y]))Z] \\ &= \mathbb{E}[f(X+\mathbb{E}[Y])]\,\mathbb{E}[Z] + \mathbb{E}[(f(X+Y) - f(X+\mathbb{E}[Y]))Z] \\ &= \mathbb{E}[f(X+Y)]\,\mathbb{E}[Z] + \mathbb{E}[f(X+\mathbb{E}[Y]) - f(X+Y)]\,\mathbb{E}[Z] + \mathbb{E}[(f(X+Y) - f(X+\mathbb{E}[Y]))Z].\end{aligned}$$

This implies

$$\mathbb{E}[f(X+Y)Z] - \mathbb{E}[f(X+Y)]\,\mathbb{E}[Z] = \mathbb{E}[f(X+\mathbb{E}[Y]) - f(X+Y)]\,\mathbb{E}[Z] + \mathbb{E}[(f(X+Y) - f(X+\mathbb{E}[Y]))Z].$$

Now, using Lipschitzness of $f$, we deterministically have that $|f(X+\mathbb{E}[Y]) - f(X+Y)| \leqslant L|Y - \mathbb{E}[Y]| = L|\mathbf{zm}(Y)|$. Similarly, $|(f(X+Y) - f(X+\mathbb{E}[Y]))Z| \leqslant L|\mathbf{zm}(Y)Z|$. Taking absolute values, we arrive at

$$|\,\mathbb{E}[f(X+\mathbb{E}[Y]) - f(X+Y)]\,\mathbb{E}[Z] + \mathbb{E}[(f(X+Y) - f(X+\mathbb{E}[Y]))Z]| \leqslant L\,\mathbb{E}[|\mathbf{zm}(Y)|(|Z| + |\,\mathbb{E}[Z]|)].$$

This immediately implies (8.21). If $|Z| \leqslant 1$ almost surely, we have $\mathbb{E}[|\mathbf{zm}(Y)Z|] \leqslant \mathbb{E}[|\mathbf{zm}(Y)|]$ and $|\,\mathbb{E}[Z]| \leqslant 1$ which yields the $2L\,\mathbb{E}[|\mathbf{zm}(Y)|]$ upper bound.

8.3.1.2. Proof of Lemma 8.9 Informally, this lemma is obvious via the tree visualization. To formally prove this lemma, we use an induction argument. For $\bar{\boldsymbol{h}}^{(1)}$, the result is trivial because $\bar{\boldsymbol{h}}_i^{(1)} = \langle \boldsymbol{k}^{(1)}, \boldsymbol{x}(i) \rangle$ which is a weighted sum of entries corresponding to $\text{set}_1(i)$. Suppose the claim holds for all layers less

than or equal to $\ell - 1$ and $\bar{\boldsymbol{h}}_i^{(\ell-1)} = f_{\ell-1}(\boldsymbol{x}_{\mathrm{set}_{\ell-1}(i)})$. For layer $\ell$, we can use the fact that $\mathrm{set}_\ell(i) = \bigcup_{j=1}^{d_\ell} \mathrm{set}_{\ell-1}((i-1)d_\ell + j)$ to conclude that

$$\begin{aligned}\bar{\boldsymbol{h}}_i^{(\ell)} &= \sum_{j=1}^{d_\ell} \boldsymbol{k}_j^{(\ell)} \sigma_{\ell-1}(\bar{\boldsymbol{h}}_{(i-1)d_\ell+j}^{(\ell-1)}) \\ &= \sum_{j=1}^{d_\ell} \boldsymbol{k}_j^{(\ell)} \sigma_{\ell-1}(f_{\ell-1}(\boldsymbol{x}_{\mathrm{set}_{\ell-1}((i-1)d_\ell+j)})) := f_\ell(\boldsymbol{x}_{\mathrm{set}_\ell(i)}).\end{aligned}$$

The latter is clearly a deterministic function of $\boldsymbol{x}_{\mathrm{set}_\ell(i)}$. Also it is independent of entry $i$ because it simply chunks the vector $\boldsymbol{x}_{\mathrm{set}_\ell(i)}$ into $d_\ell$ sub-vectors and returns a sum of weighted functions of these sub-vectors. Here, the weights are the entries of $\boldsymbol{k}^{(\ell)}$ and the functions are given by $\sigma_{\ell-1}(f_{\ell-1}(\cdot))$ (also note that the activation output is simply $\boldsymbol{h}_i^{(\ell)} = \sigma_\ell(f_\ell(\boldsymbol{x}_{\mathrm{set}_\ell(i)}))$).

### 8.4 *Proofs for learning the convolutional kernels (Proof of Theorem 4.9)*

The first part of the theorem follows trivially by combining Theorems 4.2 and 4.7. To translate a bound on the tensor spectral norm of $\boldsymbol{T}_n - \boldsymbol{L}_{\mathrm{CNN}}$ to a bound on learning the kernels, it requires a perturbation argument for tensor decompositions. This is the subject of the next lemma.

LEMMA 8.10 Let $\boldsymbol{L} = \gamma \bigotimes_{\ell=1}^{D} \boldsymbol{v}_\ell$ be a rank one tensor with $\{\boldsymbol{v}_\ell\}_{\ell=1}^D$ vectors of unit norm. Also assume $\boldsymbol{E}$ is a perturbation tensor obeying $|||\boldsymbol{E}||| \leqslant \delta$. Set

$$\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_D = \underset{\bar{\boldsymbol{u}}_1, \bar{\boldsymbol{u}}_2, \ldots, \bar{\boldsymbol{u}}_D}{\arg\max} \left\langle \boldsymbol{L}, \bigotimes_{\ell=1}^{D} \bar{\boldsymbol{u}}_\ell \right\rangle \quad \text{subject to} \quad \left\|\bar{\boldsymbol{u}}_1\right\|_{\ell_2} = \left\|\bar{\boldsymbol{u}}_2\right\|_{\ell_2} = \ldots = \left\|\bar{\boldsymbol{u}}_D\right\|_{\ell_2} = 1. \quad (8.23)$$

Then we have

$$\prod_{i=1}^{D} |\boldsymbol{u}_i^* \boldsymbol{v}_i| \geqslant 1 - 2\delta/\gamma.$$

The proof of Theorem 4.9 is complete by applying Lemma 8.10 above with $\boldsymbol{v}_\ell = \boldsymbol{k}^{(\ell)}$, $\boldsymbol{u}_\ell = \hat{\boldsymbol{k}}^{(\ell)}$, $\gamma = \alpha_{\mathrm{CNN}}$ and $\boldsymbol{E} = \boldsymbol{T}_n - \boldsymbol{L}_{\mathrm{CNN}}$. All that remains is to prove Lemma 8.10 which is the subject of the next section.

#### 8.4.1 *Proof of Lemma 8.10*

To prove this lemma, first note that for any two rank one tensors we have

$$\left\langle \bigotimes_{\ell=1}^{D} \boldsymbol{u}_\ell, \bigotimes_{\ell=1}^{D} \boldsymbol{v}_\ell \right\rangle = \prod_{i=1}^{D} \langle \boldsymbol{u}_i, \boldsymbol{v}_i \rangle.$$

Using this equality together with the fact that the vectors $\{\boldsymbol{u}_\ell\}_{\ell=1}^D$ are a maximizer for (8.23), we conclude that

$$\begin{aligned}\left\langle \boldsymbol{L}+\boldsymbol{E}, \bigotimes_{\ell=1}^{D} \boldsymbol{v}_\ell \right\rangle &\leqslant \left\langle \boldsymbol{L}+\boldsymbol{E}, \bigotimes_{\ell=1}^{D} \boldsymbol{u}_\ell \right\rangle, \\ &\leqslant \left| \left\langle \boldsymbol{L}, \bigotimes_{\ell=1}^{D} \boldsymbol{u}_\ell \right\rangle \right| + \left| \left\langle \boldsymbol{E}, \bigotimes_{\ell=1}^{D} \boldsymbol{u}_\ell \right\rangle \right|, \\ &\leqslant \gamma \prod_{i=1}^{D} |\left\langle \boldsymbol{u}_i, \boldsymbol{v}_i \right\rangle| + \delta. \end{aligned} \tag{8.24}$$

Furthermore, note that

$$\left\langle \boldsymbol{L}+\boldsymbol{E}, \bigotimes_{\ell=1}^{D} \boldsymbol{v}_\ell \right\rangle = \gamma + \left\langle \boldsymbol{E}, \bigotimes_{\ell=1}^{D} \boldsymbol{v}_\ell \right\rangle \geqslant \gamma - \delta. \tag{8.25}$$

Combining (8.24) and (8.25), we conclude that

$$\gamma \prod_{i=1}^{D} |\left\langle \boldsymbol{u}_i, \boldsymbol{v}_i \right\rangle| + \delta \geqslant \gamma - \delta,$$

concluding the proof.

## 9. Lower bounds on CNN gain

LEMMA 9.1 Consider a CNN model per Section 2 with $f_{\text{CNN}}()$ the corresponding CNN function per (2.2). We have the following lower bound on the nonlinearity parameter $\alpha_{\text{CNN}} = \prod_{\ell=1}^{D} \mathbb{E}[\phi_\ell'(\tilde{\boldsymbol{h}}_1^{(\ell)})]$ from Definition 4.3.

- ReLU model: $\phi(x) = \max(0, x)$ with the added assumption that the kernels have mean larger than zero and are modestly diffused. Specifically, assume

$$\frac{(\mathbf{1}^T \boldsymbol{k}_r)}{\|\boldsymbol{k}_r\|_{\ell_2}} \geqslant 4. \tag{9.1}$$

  Then

$$\alpha_{\text{CNN}} \geqslant \frac{1}{4}.$$

- Softplus model: $\phi(x) = \log(1 + e^x)$ with the added assumption that the kernels have mean larger than zero, are modestly diffused and have a sufficiently large Euclidean norm. Specifically,

assume

$$\frac{(\mathbf{1}^T \boldsymbol{k}_r)}{\|\boldsymbol{k}_r\|_{\ell_2}} \geqslant 10 \quad \text{and} \quad \|\boldsymbol{k}_\ell\|_{\ell_2} \geqslant 1. \tag{9.2}$$

Then

$$\alpha_{\text{CNN}} \geqslant 0.3.$$

*Proof.* **ReLU:** in this case note that

$$\begin{aligned}
\alpha_{\text{CNN}} &= \prod_{\ell=1}^{D} \mathbb{E}\left[\phi'\left(\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}\right)\right] \\
&= \prod_{\ell=1}^{D} \mathbb{E}\left[\mathbb{1}_{\left\{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)} \geqslant 0\right\}}\right] \\
&= \prod_{\ell=1}^{D} \mathbb{P}\left\{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)} \geqslant 0\right\}.
\end{aligned}$$

Thus, using $t = \mathbb{E}\left[\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}\right]$, we arrive at

$$\mathbb{P}\left\{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)} < 0\right\} \leqslant e^{-\frac{\left(\mathbb{E}\left[\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}\right]\right)^2}{2\prod_{r=1}^{\ell}\|\boldsymbol{k}_r\|_{\ell_2}^2}}. \tag{9.3}$$

Since the entries of $\boldsymbol{h}^{(\ell)}$ and $\bar{\boldsymbol{h}}^{(\ell)}$ are i.i.d., we use $H^{(\ell)}$ and $\bar{H}^{(\ell)}$ to denote the corresponding distributions. We note that

$$\begin{aligned}
\mathbb{E}\left[\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}\right] &= \mathbb{E}\left[\langle \boldsymbol{k}^{(\ell)}, \boldsymbol{h}^{(\ell)}[\boldsymbol{i}_\ell]\rangle\right] \\
&= \left(\mathbf{1}^T \boldsymbol{k}^{(\ell)}\right) \mathbb{E}[H^{(\ell)}] \\
&= \left(\mathbf{1}^T \boldsymbol{k}^{(\ell)}\right) \mathbb{E}\left[\phi\left(\bar{H}^{(\ell)}\right)\right] \\
&\geqslant \left(\mathbf{1}^T \boldsymbol{k}^{(\ell)}\right) \phi\left(\mathbb{E}[\bar{H}^{(\ell)}]\right) \\
&= \left(\mathbf{1}^T \boldsymbol{k}^{(\ell)}\right) \cdot \mathbb{E}[\bar{H}^{(\ell)}],
\end{aligned}$$

where in the last inequality we used the fact that $\left(\mathbf{1}^T \boldsymbol{k}^{(\ell)}\right) \geqslant 0$ and applied Jenson's inequality for a convex $\phi$. Applying this inequality recursively, we arrive at

$$\mathbb{E}\left[\bar{\boldsymbol{h}}_{i_\ell}^{(\ell)}\right] \geqslant \prod_{r=1}^{\ell} \left(\mathbf{1}^T \boldsymbol{k}^{(r)}\right) \mathbb{E}[\phi(g)] = \frac{1}{\sqrt{2\pi}} \prod_{r=1}^{\ell} \left(\mathbf{1}^T \boldsymbol{k}^{(r)}\right). \tag{9.4}$$

Using the latter in (9.3), we arrive at

$$\mathbb{P}\left\{\bar{\boldsymbol{h}}_{i_\ell}^{(\ell)} < 0\right\} \leqslant e^{-\frac{1}{4\pi}\prod_{r=1}^{\ell}\frac{\left(\mathbf{1}^T \boldsymbol{k}^{(r)}\right)^2}{\|\boldsymbol{k}^{(r)}\|_{\ell_2}^2}} \quad \Rightarrow \quad \mathbb{P}\left\{\bar{\boldsymbol{h}}_{i_\ell}^{(\ell)} \geqslant 0\right\} \geqslant 1 - e^{-\frac{1}{4\pi}\prod_{r=1}^{\ell}\frac{\left(\mathbf{1}^T \boldsymbol{k}^{(r)}\right)^2}{\|\boldsymbol{k}^{(r)}\|_{\ell_2}^2}}.$$

Thus, using the diffusion assumption (9.2) in the latter inequality, we arrive at

$$\alpha_{\mathrm{CNN}} \geqslant \prod_{\ell=1}^{D} \left(1 - e^{-\frac{1}{4\pi}\nu^{2\ell}}\right).$$

Thus, using the fact that for $0 \leqslant x_1, x_2, ..., x_n \leqslant 1$, we have

$$\prod_{i=1}^{n}(1 - x_i) \geqslant 1 - \sum_{i=1}^{n} x_i,$$

we conclude that

$$\begin{aligned}
\alpha_{\mathrm{CNN}} \geqslant & 1 - \sum_{\ell=1}^{D} e^{-\frac{1}{4\pi}\nu^{2\ell}} \\
\geqslant & 1 - \int_0^D e^{-\frac{1}{4\pi}\nu^{2x}}\,\mathrm{d}x \\
\geqslant & 1 - \int_0^\infty e^{-\frac{1}{4\pi}\nu^{2x}}\,\mathrm{d}x \\
= & 1 - \frac{E_1(\frac{1}{4\pi})}{\log \nu^2} \\
= & 1 - \frac{E_1(\frac{1}{4\pi})}{\log 16} \\
\geqslant & \frac{1}{4}.
\end{aligned}$$

In the above, $E_1(x) = \int_1^\infty \frac{e^{-tx}}{t}\,\mathrm{d}t$ and we used the fact that $E_1\left(\frac{1}{4\pi}\right) \leqslant 1.016$ and $\nu \geqslant 4$. Thus,

$$\alpha_{\mathrm{CNN}} \geqslant \frac{1}{4}.$$

Softplus: for the softplus activation, we have

$$\begin{aligned}
\alpha_{\mathrm{CNN}} &= \prod_{\ell=1}^{D} \mathbb{E}\left[\phi'\left(\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}\right)\right] \\
&= \prod_{\ell=1}^{D} \mathbb{E}\left[\frac{1}{1+e^{-\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}}}\right] \\
&= \prod_{\ell=1}^{D} \mathbb{E}\left[\left(1-\frac{1}{1+e^{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}}}\right)\right] \\
&= \prod_{\ell=1}^{D}\left(1-\mathbb{E}\left[\frac{1}{1+e^{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}}}\right]\right).
\end{aligned}$$

Thus, using the fact that for $0 \leqslant x_1, x_2, ..., x_n \leqslant 1$, we have

$$\prod_{i=1}^{n}(1-x_i) \geqslant 1-\sum_{i=1}^{n} x_i,$$

we conclude that

$$\begin{aligned}
\alpha_{\mathrm{CNN}} \geqslant & 1-\sum_{\ell=1}^{D}\mathbb{E}\left[\frac{1}{1+e^{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}}}\right] \\
\geqslant & 1-\sum_{\ell=1}^{D}\mathbb{E}\left[\frac{\mathbb{1}\left\{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)} < \mathbb{E}[\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}]/2\right\} + \mathbb{1}\left\{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)} \geqslant \mathbb{E}[\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}]/2\right\}}{1+e^{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}}}\right] \\
\geqslant & 1-\sum_{\ell=1}^{D}\mathbb{E}\left[\mathbb{1}\left\{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)} < \mathbb{E}[\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}]/2\right\}\right] - \sum_{\ell=1}^{D}\mathbb{E}\left[\frac{\mathbb{1}\left\{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)} \geqslant \mathbb{E}[\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}]/2\right\}}{1+e^{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}}}\right] \\
= & 1-\sum_{\ell=1}^{D}\mathbb{P}\left\{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)} < \mathbb{E}[\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}]/2\right\} - \sum_{\ell=1}^{D}\mathbb{E}\left[\frac{\mathbb{1}\left\{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)} \geqslant \mathbb{E}[\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}]/2\right\}}{1+e^{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}}}\right] \\
\geqslant & 1-\sum_{\ell=1}^{D}\mathbb{P}\left\{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)} < \mathbb{E}[\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}]/2\right\} - \sum_{\ell=1}^{D}\frac{1}{1+e^{\mathbb{E}\left[\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}\right]/2}}. \qquad (9.5)
\end{aligned}$$

We bound the expected value of the hidden units similar to the argument for ReLU activations. The only difference is that in the identity (9.4) we need to use the softplus activation in lieu of the ReLU activation for $\phi$. Therefore, (9.4) changes to

$$\mathbb{E}\left[\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}\right] \geqslant \prod_{r=1}^{\ell}\left(\mathbf{1}^T\boldsymbol{k}^{(r)}\right)\mathbb{E}[\phi(g)] = 0.806059\prod_{r=1}^{\ell}\left(\mathbf{1}^T\boldsymbol{k}^{(r)}\right). \tag{9.6}$$

Similar to the ReLU argument, we note that $\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}$ is a Lipschitz function of a Gaussian random vector $(\boldsymbol{g})$ with Lipschitz constant equal to $\prod_{r=1}^{\ell}\left\|\boldsymbol{k}^{(r)}\right\|_{\ell_2}$. Using Lipschitz concentration of Gaussians, we thus have

$$\mathbb{P}\left\{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)} - \mathbb{E}\left[\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}\right] < -t\right\} \leqslant e^{-\frac{t^2}{2\prod_{r=1}^{\ell}\left\|\boldsymbol{k}^{(r)}\right\|_{\ell_2}}}. \tag{9.7}$$

Thus, using $t = \mathbb{E}\left[\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}\right]/2$, we arrive at

$$\mathbb{P}\left\{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)} < \mathbb{E}[\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}]/2\right\} \leqslant e^{-\frac{\left(\mathbb{E}\left[\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}\right]\right)^2}{8\prod_{r=1}^{\ell}\|\boldsymbol{k}_r\|_{\ell_2}^2}}. \tag{9.8}$$

Thus, using the diffusion assumption (9.2) with $\nu \geqslant 4$, we have

$$\begin{aligned}\sum_{\ell=1}^{D-1}\mathbb{P}\left\{\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)} < \mathbb{E}[\bar{\boldsymbol{h}}_{\boldsymbol{i}_\ell}^{(\ell)}]/2\right\} &\leqslant \sum_{\ell=1}^{D-1} e^{-0.1\nu^{2\ell}} \\ &\leqslant \int_0^{D-1} e^{-0.1\nu^{2x}}\,\mathrm{d}x \\ &\leqslant \int_0^{+\infty} e^{-0.1\nu^{2x}}\,\mathrm{d}x \\ &= \frac{E_1(0.1)}{\log\nu^2}.\end{aligned}$$

Also using (9.6) and assuming $\left\|\boldsymbol{k}^{(\ell)}\right\|_{\ell_2} \geqslant 1$, we have

$$\begin{aligned}
\sum_{\ell=1}^{D-1} \frac{1}{1+e^{\mathbb{E}\left[\tilde{\boldsymbol{h}}_{i_\ell}^{(\ell)}\right]/2}} &\leqslant \sum_{\ell=1}^{D-1} \frac{1}{1+e^{0.4030295\prod_{r=1}^{\ell}\left(\mathbf{1}^T\boldsymbol{k}^{(r)}\right)}} \\
&\leqslant \sum_{\ell=1}^{D-1} e^{-0.4030295\prod_{r=1}^{\ell}\left(\mathbf{1}^T\boldsymbol{k}^{(r)}\right)} \\
&\leqslant \sum_{\ell=1}^{D-1} e^{-0.4030295\cdot\nu^{\ell}\cdot\prod_{r=1}^{\ell}\left\|\boldsymbol{k}^{(r)}\right\|_{\ell_2}} \\
&\leqslant \sum_{\ell=1}^{D-1} e^{-0.4030295\cdot\nu^{\ell}} \\
&\leqslant \int_0^{D-1} e^{-0.4030295\cdot\nu^{x}} \\
&\leqslant \int_0^{+\infty} e^{-0.4030295\cdot\nu^{x}} \\
&= \frac{E_1(0.4030295)}{\log\nu}.
\end{aligned}$$

Plugging the latter two inequalities in (9.5) allows to conclude that for $\nu \geqslant 10$

$$\begin{aligned}
\alpha_{\mathrm{CNN}} &= 1 - \frac{E_1(0.1)}{\log\nu^2} - \frac{E_1(0.4030295)}{\log\nu} \\
&\geqslant 0.3.
\end{aligned}$$

□

## Data Availability Statement

No new data were generated or analyzed in support of this research.


## Funding

National Science Foundation (1932254 and 2046816 to S.O., 1846369 and 1813877 to M.S.); Packard Fellowship in Science and Engineering (M.S.); Sloan Research Fellowship in Mathematics (M.S.); Air Force Office of Scientific Research Young Investigator Program (FA9550-18-1-0078 to M.S.); DARPA Learning with Less Labels and Fast Network Interface Cards Programs (M.S.); Google Faculty Research Award (M.S.).

## References


Allen-Zhu, Z., Li, Y. & Song, Z. (2019) A convergence theory for deep learning via over-parameterization. *International Conference on Machine Learning*. PMLR, pp. 242–252.

Anandkumar, A., Ge, R., Hsu, D., Kakade, S. M. & Telgarsky, M. (2014) Tensor decompositions for learning latent variable models. *J. Mach. Learn. Res.*, **15**, 2773–2832.

Anandkumar, A., Ge, R. & Janzamin, M. (2014) Guaranteed non-orthogonal tensor decomposition via alternating rank-1 updates (in press). arXiv:1402.5180.

Arora, S., Bhaskara, A., Ge, R. & Ma, T. (2014) Provable bounds for learning some deep representations. *International Conference on Machine Learning*, PMLR. pp. 584–592.

Baldi, P. & Vershynin, R. (2019) The capacity of feedforward neural networks. *Neural Netw.*, **116**, 288–311.

Bartlett, P. L., Foster, D. J. & Telgarsky, M. J. (2017) Spectrally-normalized margin bounds for neural networks. *Advances in Neural Information Processing Systems*, NeurIPS, pp. 6240–6249.

Belkin, M., Hsu, D., Ma, S. & Mandal, S. (2018) Reconciling modern machine learning and the bias-variance trade-off. *Statistics*, **1050**, 28.

Bro, R. (1997) PARAFAC. Tutorial and applications. *Chemom. Intell. Lab. Syst.*, **38**, 149–171.

Brutzkus, A. & Globerson, A. (2017) Globally optimal gradient descent for a ConvNet with Gaussian inputs. *Proceedings of the 34th International Conference on Machine Learning*, PMLR, vol. 70, pp. 605–614.

Chizat, L. & Bach, F. (2018) On the global convergence of gradient descent for over-parameterized models using optimal transport. *Advances in Neural Information Processing Systems*, NeurIPS, vol. 31, pp. 3036–3046.

Cohen, N., Sharir, O. & Shashua, A. (2016) On the expressive power of deep learning: a tensor analysis. *Conference on Learning Theory*, PMLR, pp. 698–728.

Cohen, N. & Shashua, A. (2016) Convolutional rectifier networks as generalized tensor decompositions. *International Conference on Machine Learning*, PMLR, pp. 955–963.

Collobert, R. & Weston, J. (2008) A unified architecture for natural language processing: deep neural networks with multitask learning. *Proceedings of the 25th International Conference on Machine Learning*. ACM, PMLR, pp. 160–167.

Dirksen, S. (2015) Tail bounds via generic chaining. *Electronic Journal of Probability,* The Institute of Mathematical Statistics and the Bernoulli Society, vol. 15.

Du, S., Lee, J., Tian, Y., Singh, A. & Poczos, B. (2018) Gradient descent learns one-hidden-layer CNN: don't be afraid of spurious local minima. *International Conference on Machine Learning*. PMLR, pp. 1339–1348.

Du, Simon, *et al.* (2019) Gradient descent finds global minima of deep neural networks. *International Conference on Machine Learning*, PMLR.

Du, S. S., Lee, J. D. & Tian, Y. (2018) When is a convolutional filter easy to learn? *6th International Conference on Learning Representations, ICLR 2018*.

Fu, H., Chi, Y. & Liang, Y. (2020) Guaranteed recovery of one-hidden-layer neural networks via cross entropy. *IEEE Trans. Signal Process.*, **68**, 3225–3235.

Ge, R., Huang, F., Jin, C. & Yuan, Y. (2015) Escaping from saddle points—online stochastic gradient for tensor decomposition. *Conference on Learning Theory*, PMLR, pp. 797–842.

Ge, R., Lee, J. D. & Ma, T. (2018) Learning one-hidden-layer neural networks with landscape design. *International Conference on Learning Representations*, ICLR.

Goel, S., Kanade, V., Klivans, A. & Thaler, J. (2017) Reliably learning the relu in polynomial time. *Conference on Learning Theory*. PMLR, pp. 1004–1042.

Goel, S., Klivans, A. & Meka, R. (2018) Learning one convolutional layer with overlapping patches. *International Conference on Machine Learning*, PMLR, pp. 1783–1791.

He, K., Zhang, X., Ren, S. & Sun, J. (2016) Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778.

Janzamin, M., Sedghi, H. & Anandkumar, A. (2015) Beating the perils of non-convexity: guaranteed training of neural networks using tensor methods (in press). arXiv:1506.08473.

Ji, Z. & Telgarsky, M. J. (2019) Gradient descent aligns the layers of deep linear networks. *7th International Conference on Learning Representations, ICLR 2019*.

Kossaifi, J., Lipton, Z. C., Kolbeinsson, A., Khanna, A., Furlanello, T. & Anandkumar, A. (2020) Tensor regression networks. *J. Mach. Learn. Res.*, JMLR, **21**, 1–21.

Kossaifi, J., Panagakis, Y., Anandkumar, A. & Pantic, M. (2019) Tensorly: tensor learning in python. *J. Mach. Learn. Res.*, JMLR, **20**, 925–930.

Krizhevsky, A., Sutskever, I. & Hinton, G. E. (2012) Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, NeurIPS, pp. 1097–1105.

Li, Y. & Yuan, Y. (2017) Convergence analysis of two-layer neural networks with relu activation. *Advances in Neural Information Processing Systems*, NeurIPS, pp. 597–607.

Malach, E. & Shalev-Shwartz, S. (2018) A provably correct algorithm for deep learning that actually works (in press). arXiv:1803.09522.

Mei, S., Montanari, A. & Nguyen, P.-M. (2018) A mean field view of the landscape of two-layer neural networks. *Proc. Natl. Acad. Sci. U. S. A.*, **115**, E7665–E7671.

Mondelli, M. & Montanari, A. (2018) On the connection between learning two-layers neural networks and tensor decomposition. *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 2019.

Neyshabur, B., Bhojanapalli, S. & Srebro, N. (2018) A PAC-Bayesian approach to spectrally-normalized margin bounds for neural networks. *International Conference on Learning Representations*, ICLR.

Neyshabur, B. & Li, Z. (2019) Towards understanding the role of over-parametrization in generalization of neural networks. *International Conference on Learning Representations (ICLR)*.

Oymak, S. (2018) Learning compact neural networks with regularization. *International Conference on Machine Learning*, PMLR, pp. 3966–3975.

Oymak, S. & Soltanolkotabi, M. (2019) Overparameterized nonlinear learning: gradient descent takes the shortest path? *International Conference on Machine Learning*. PMLR, pp. 4951–4960.

Oymak, S. & Soltanolkotabi, M. (2020) Towards moderate overparameterization: global convergence guarantees for training shallow neural networks. *IEEE J. Sel. Areas Inf. Theory*, vol. 1, pp. 84–105.

Raghu, M., Poole, B., Kleinberg, J., Ganguli, S. & Sohl-Dickstein, J. (2017) On the expressive power of deep neural networks. *In international conference on machine learning*, pp. 2847–2854, PMLR.

Sagun, L., Evci, U., Guney, V. U., Dauphin, Y. & Bottou, L. (2017) Empirical analysis of the Hessian of over-parametrized neural networks. *International Conference on Learning Representations* (ICLR) 2018.

Sidiropoulos, N. D., De Lathauwer, L., Fu, X., Huang, K., Papalexakis, E. E. & Faloutsos, C. (2017) Tensor decomposition for signal processing and machine learning. *IEEE Trans. Signal Process.*, **65**, 3551–3582.

Soltanolkotabi, M. (2017) Learning relus via gradient descent. *Advances in Neural Information Processing Systems*, NeurIPS, pp. 2007–2017.

Soltanolkotabi, M., Javanmard, A. & Lee, J. D. (2018) Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *IEEE Trans. Inf. Theory*, **65**, 742–769.

Soudry, D. & Carmon, Y. (2016) No bad local minima: data independent training error guarantees for multilayer neural networks (in press). arXiv:1605.08361.

Stein, C., Diaconis, P., Holmes, S., Reinert, G., et al. (2004) Use of exchangeable pairs in the analysis of simulations. In *Stein's Method*, pp. 1–25. Institute of Mathematical Statistics.

Talagrand, M. (2006) The Generic Chaining: Upper and Lower Bounds of Stochastic Processes. Springer Science & Business Media.

Talagrand, M. (2014) Gaussian processes and the generic chaining. *Upper and Lower Bounds for Stochastic Processes*. Springer, pp. 13–73.

Tomioka, R. & Suzuki, T. (2014) Spectral norm of random tensors (in press). arXiv:1407.1870.

Van den Oord, A., Dieleman, S. & Schrauwen, B. (2013) Deep content-based music recommendation. *Advances in Neural Information Processing Systems*, NeurIPS, pp. 2643–2651.

Zhong, K., Song, Z. & Dhillon, I. S. (2017) Learning non-overlapping convolutional neural networks with multiple kernels (in press). *arXiv:1711.03440*.

ZHONG, K., SONG, Z., JAIN, P., BARTLETT, P. L. & DHILLON, I. S. (2017) Recovery guarantees for one-hidden-layer neural networks. *Proceedings of the 34th International Conference on Machine Learning*, PMLR, vol. 70, pp. 4140–4149.



## A. Centered ERM and resolving sign and scaling ambiguities

Throughout this section, we consider a single-filter CNN model with kernels $(\boldsymbol{k}^{(\ell)})_{\ell=1}^{D}$ as described in (2.1) and set $R = 1$. We note that DeepTD operates by accurately approximating the rank one tensor $\bigotimes_{\ell=1}^{D} \boldsymbol{k}^{(\ell)}$ from data. Therefore, DeepTD can only recover the convolutional kernels up to sign/scale ambiguities (SSA). In general, it may not be possible to recover the ground truth kernels from the training data. For instance, when activations are ReLU, the norms of the kernels cannot be estimated from data as multiplying a kernel and dividing another by the same positive scalar leads to the same training data. However, we can try to learn a good approximation $\hat{f}_{\mathrm{CNN}}()$ of the network $f_{\mathrm{CNN}}()$ to minimize the risk $\mathbb{E}[(f_{\mathrm{CNN}}(\boldsymbol{x}) - \hat{f}_{\mathrm{CNN}}(\boldsymbol{x}))^2]$.

To this aim, we introduce centered empirical risk minimization (CERM) which is a slight modification of ERM. Let us first describe how finding a good $\hat{f}_{\mathrm{CNN}}()$ can be formulated with CERM. Given $n$ i.i.d. data points $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n} \sim (\boldsymbol{x}, y)$, and a function class $\mathscr{F}$, CERM applies ERM after centering the residuals. Given $f \in \mathscr{F}$, define the average residual function $r_{\mathrm{avg}}(f) = \frac{1}{n}\sum_{i=1}^{n} y_i - f(\boldsymbol{x}_i)$. We define the CERM as

$$\begin{aligned}\hat{f} &= \min_{f\in\mathscr{F}} \frac{1}{n}\sum_{i=1}^{n}(y_i - f(\boldsymbol{x}_i) - r_{\mathrm{avg}}(f))^2,\\ &= \min_{f\in\mathscr{F}} \frac{1}{n}\sum_{i=1}^{n}\big(y_i - f(\boldsymbol{x}_i) - \mathbb{E}[(y_i - f(\boldsymbol{x}_i))]\big)^2 - \frac{1}{n^2}\left(\sum_{i=1}^{n} y_i - f(\boldsymbol{x}_i) - \mathbb{E}[(y_i - f(\boldsymbol{x}_i))]\right)^2. \qquad \text{(A.1)}\end{aligned}$$

The remarkable benefit of CERM over ERM is the fact that the learning rate does not suffer from the label or function bias. This is in similar nature to the DeepTD algorithm that applies label centering. In the proofs (in particular Section A.2, Theorem A.2), we provide a generalization bound on the CERM solution (A.1) in terms of the Lipschitz covering number of the function space. While (A.1) can be used to learn all kernels, it does not provide an efficient algorithm. Instead, we will use CERM to resolve SSA after estimating the kernels via DeepTD. Interestingly, this approach only requires a few ($\mathscr{O}(D)$) extra training samples. Inspired from CERM, in Section A.1, we propose a greedy algorithm to address SSA. We will apply CERM to the following function class with bounded kernels,

$$\mathscr{F}_{\hat{\boldsymbol{k}},B} := \Big\{f : \mathbb{R}^p \mapsto \mathbb{R}\Big|\text{f is a CNN function of the form (2.2) with kernels } \{\beta_\ell \hat{\boldsymbol{k}}^{(\ell)}\}_{\ell=1}^{D} \text{ with } |\beta_\ell| \leqslant B^{\frac{1}{D}}\Big\}. \qquad \text{(A.2)}$$

In words, this is the function class of all CNN functions with kernels the same as those obtained by DeepTD up to sign/scale ambiguities $\{\beta_\ell\}_{\ell=1}^{D}$ where the maximum scale ambiguity is $B$.

THEOREM A.1 Let $f_{\mathrm{CNN}}()$ be defined via (2.2) with convolutional kernels $\{\boldsymbol{k}^{(\ell)}\}_{\ell=1}^{D}$ obeying $\|\boldsymbol{k}^{(\ell)}\|_{\ell_2} \leqslant B^{1/D}$ for some $B > 0$ and consider the function class $\mathscr{F}_{\hat{\boldsymbol{k}},B}$ above with the same choice of $B$. Assume we have $n$ i.i.d. samples $(\boldsymbol{x}_i, y_i) \sim (\boldsymbol{x}, y)$ where $\boldsymbol{x} \sim \mathscr{N}(0, \boldsymbol{I}_p)$ and $y = f_{\mathrm{CNN}}(\boldsymbol{x})$. Suppose for some $\varepsilon \leqslant B$,

$$n \geqslant cB^2 D \log\left(\frac{CDBp}{\epsilon}\right) \max\left(\frac{1}{\epsilon}, \frac{1}{\epsilon^2}\right)$$

holds for fixed numerical constants $c, C > 0$. Then the solution $\hat{f}$ to the CERM problem (A.1) obeys

$$\mathbb{E}\left[\left(\hat{f}(\boldsymbol{x}) - f_{\mathrm{CNN}}(\boldsymbol{x})\right)^2\right] \leqslant \min_{f \in \mathscr{F}_{\hat{\boldsymbol{k}},B}} \mathbb{E}[(f(\boldsymbol{x}) - f_{\mathrm{CNN}}(\boldsymbol{x}))^2] + \varepsilon \tag{A.3}$$

on a new sample $\boldsymbol{x} \in \mathscr{N}(0, \boldsymbol{I}_p)$ with probability at least $1 - e^{-\gamma n} - 4n\exp(-p)$ with $\gamma > 0$ an absolute constant.

The above theorem states that CERM finds the sign/scale ambiguity that accurate estimates the labels on new data as long as the number of samples which are used in CERM exceeds the depth of the network by constant/log factors. In the next section, we present a greedy heuristic for finding the CERM estimate.

**Algorithm 1** Greedily algorithm for resolving sign/scale ambiguities for Leaky ReLU activations.

**procedure** MAXCORR
**Inputs:** Data $(y_i, \boldsymbol{x}_i)_{i=1}^n$, estimates $\{\hat{\boldsymbol{k}}^{(\ell)}\}_{\ell=1}^D$.
$\rho_{\max} \leftarrow |\mathrm{Corr}(\{\hat{\boldsymbol{k}}^{(\ell)}\}_{\ell=1}^D, 0)|$, FLIP$\leftarrow$TRUE.
**While** FLIP
  FLIP$\leftarrow$FALSE.
  **For** $1 \leqslant \ell \leqslant D$
    $\rho \leftarrow |\mathrm{Corr}(\{\hat{\boldsymbol{k}}^{(1)}, \ldots, -\hat{\boldsymbol{k}}^{(\ell)}, \ldots, \hat{\boldsymbol{k}}^{(D)}\}, 0)|$.
    **If** $\rho > \rho_{\max}$
      $\rho_{\max} \leftarrow \rho$
      $\hat{\boldsymbol{k}}^{(\ell)} \leftarrow -\hat{\boldsymbol{k}}^{(\ell)}$
      FLIP$\leftarrow$TRUE
    **EndIf**
  **EndFor**
**EndWhile**
$\gamma \leftarrow \mathrm{Corr}(\{\hat{\boldsymbol{k}}^{(\ell)}\}_{\ell=1}^D, 1)$.
**return** kernels $\{\hat{\boldsymbol{k}}^{(\ell)}\}_{\ell=1}^D$, scaling $\gamma$.
**end procedure**

### A.1 *Greedy algorithm for resolving sign and scale ambiguities*

In order to resolve SSA, inspired from CERM, we propose Algorithm 1 which operates over the function class,

$$\mathscr{F}_{\hat{\boldsymbol{k}}} := \left\{\gamma f : \mathbb{R}^p \mapsto \mathbb{R} \middle| \text{f is a CNN of the form (2.2) with kernels} \{\beta_\ell \hat{\boldsymbol{k}}^{(\ell)}\}_{\ell=1}^{D} \text{ with } \beta_\ell \in \{1,-1\},\ \gamma \geqslant 0\right\}. \tag{A.4}$$

It first determines the signs $\beta_\ell$ by locally optimizing the kernels and then finds a global scaling $\gamma > 0$. In the first phase, the algorithm attempts to maximize the correlation between the centered labels $y_{c,i} = y_i - n^{-1}\sum_{i=1}^n y_i$ and the $\hat{f}_{\text{CNN}}()$ predictions given by $\hat{y}_{c,i} = \hat{y}_i - n^{-1}\sum_{i=1}^n \hat{y}_i$. It goes over all kernels one by one and it flips a kernel ($\hat{\boldsymbol{k}}^{(\ell)} \to -\hat{\boldsymbol{k}}^{(\ell)}$) if flipping increases the correlation. This process goes on as long as there is an improvement. Afterwards, we use a simple linear regression to get the best scaling $\gamma$ by minimizing the centered empirical loss $\sum_{i=1}^n (y_{c,i} - \gamma \hat{y}_{c,i})^2$. While our approach is applicable to arbitrary activations, it is tailored towards homogeneous activations ($\phi(cx) = c\phi(x)$). The reason is that for homogeneous activations, function classes (A.2) and (A.4) coincide and a single global scaling $\gamma$ is sufficient. Note that ReLU and the identity activation (i.e. no activation) are both homegeneous, in fact they are elements of a larger homogeneous activation family named Leaky ReLU. Leaky ReLU is parametrized by some scalar $0 \leqslant \beta \leqslant 1$ and defined as follows:

$$LReLU(x) = \begin{cases} x \text{ if } x \geqslant 0, \\ \beta x \text{ if } x < 0. \end{cases}$$

**Algorithm 2** Return the correlation between centered labels.

1: **procedure** CORR($\{\hat{\boldsymbol{k}}^{(\ell)}\}_{\ell=1}^D$, opt)

2: $\hat{y}_i \leftarrow f_{\text{CNN}}(\{\hat{\boldsymbol{k}}^{(\ell)}\}_{\ell=1}^D; \boldsymbol{x}_i)$.

3: $y_{\text{c},i} \leftarrow y_i - \frac{1}{n}\sum_{i=1}^n y_i$ and $\hat{y}_{\text{c},i} \leftarrow \hat{y}_i - \frac{1}{n}\sum_{i=1}^n \hat{y}_i$.

4: $\rho \leftarrow \sum_{i=1}^n y_{\text{c},i}\hat{y}_{\text{c},i}$.

5: **return** $\rho$ **if** opt $= 0$, $\rho/(\sum_{i=1}^n \hat{y}_{\text{c},i}^2)$ **if** opt $= 1$.

6: **end procedure**

### A.2 *Generalization bounds for CERM*

In this section, we prove a generalization bound for CERM (A.1). The following theorem shows that using a finite sample size $n$, CERM is guaranteed to choose a function close to population's minimizer. For the sake of this section, $\|f\|_{L_\infty}$ will be the Lipschitz constant of a function.

THEOREM A.2 Let $\mathscr{F}$ be a class of functions $f : \mathbb{R}^p \to \mathbb{R}$. Suppose $\sup_{f\in\mathscr{F}} \|f\|_{L_\infty} \leqslant R$. Let $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n \sim (\boldsymbol{x}, y)$ be i.i.d. data points where $\boldsymbol{x} \sim \mathcal{N}(0, \boldsymbol{I}_p)$ and $y$ is so that $y - \mathbb{E}[y]$ is $K$ subgaussian. Suppose $\mathscr{F}$ has $\|\cdot\|_{L_\infty}$, $\delta$-covering bound obeying $\log N_\delta \leqslant s \log \frac{C}{\delta}$ for some constants $s \geqslant 1, C \geqslant R$. Given $\varepsilon \leqslant \bar{K} = K + R$, suppose $n \geqslant \mathscr{O}(\max\{\varepsilon^{-1}, \varepsilon^{-2}\}\bar{K}^2 s \log \frac{C' p\bar{K}}{\varepsilon})$ for some $C' > 0$. Then the CERM

output (A.1) obeys

$$\mathbb{E}[\left(\hat{f}(\boldsymbol{x}) - y - \mathbb{E}[(\hat{f}(\boldsymbol{x}) - y)]\right)^2] \leqslant \min_{f\in\mathscr{F}} \mathbb{E}[(f(\boldsymbol{x}) - y - \mathbb{E}[(f(\boldsymbol{x}) - y)])^2] + \varepsilon \tag{A.5}$$

with probability $1 - \exp(-\mathscr{O}(n)) - 4n\exp(-p)$.

*Proof.* Consider the centered empirical loss that can alternatively be written in the form

$$\begin{aligned} E(f) &= \frac{1}{n}\sum_{i=1}^{n} \mathbf{zm}(y_i - f(\boldsymbol{x}_i))^2 - \frac{1}{n^2}(\sum_{i=1}^{n} \mathbf{zm}(y_i - f(\boldsymbol{x}_i)))^2 - \mathbb{E}[\mathbf{zm}(f(\boldsymbol{x}) - y)^2]. \\ &= \frac{1}{n}\sum_{i=1}^{n} \mathbf{zm}(\mathbf{zm}(y_i - f(\boldsymbol{x}_i))^2) - \frac{1}{n^2}(\sum_{i=1}^{n} \mathbf{zm}(y_i - f(\boldsymbol{x}_i)))^2 \\ &:= T_1 + T_2. \end{aligned} \tag{A.6}$$

To prove the theorem, we will simply bound the supremum $\sup_{f\in\mathscr{F}} |E(f)| \leqslant \sup_{f\in\mathscr{F}} |T_1 + T_2|$. Pick a $\delta$ covering $\mathscr{F}_\delta$ of $\mathscr{F}$ with size $s\log\frac{C}{\delta}$ where $\delta$ will be determined later in this proof. We first bound $E(f)$ for all $f \in \mathscr{F}_\delta$. Given a fixed $f$, observe that $\|\mathbf{zm}(\mathbf{zm}(y_i - f(\boldsymbol{x}_i))^2)\|_{\psi_1} \leqslant \mathscr{O}(\bar{K})^2 = \mathscr{O}(K+R)^2$ which follows from $\|\mathbf{zm}(y_i - f(\boldsymbol{x}_i))\|_{\psi_2} \leqslant \mathscr{O}(\bar{K}) = \mathscr{O}(K+R)$. Applying sub-exponential concentration, since $T_1$ is sum of i.i.d. sub-exponentials, we have

$$\mathbb{P}(|T_1| \geqslant \varepsilon) \leqslant \exp(-\mathscr{O}(n\min\{\varepsilon^2/\bar{K}^2, \varepsilon/\bar{K}\})). \tag{A.7}$$

Next, since $\|\mathbf{zm}(y_i - f(\boldsymbol{x}_i))\|_{\psi_2} \leqslant \mathscr{O}(\bar{K})$, we can conclude that $\|\frac{1}{n}\sum_{i=1}^{n} \mathbf{zm}(y_i - f(\boldsymbol{x}_i))\|_{\psi_2} \leqslant \mathscr{O}(\bar{K})/\sqrt{n} \implies \|T_2\|_{\psi_1} \leqslant \mathscr{O}(\bar{K})^2/n$. Using (A.7) for $T_1$ and the sub-exponential tail bound for $T_2$ holds when $\varepsilon \leqslant \bar{K}$, and assuming the number of samples $n$ obeys $n \geqslant \mathscr{O}(\max\{\varepsilon^{-1}, \varepsilon^{-2}\}\bar{K}^2 s\log\frac{C}{\delta})$, then for all cover elements

$$|T_1| + |T_2| \leqslant 2\varepsilon$$

holds with probability at least $1 - \exp(-\mathscr{O}(n))$. To conclude the proof, we need to move from the cover $\mathscr{F}_\delta$ to $\mathscr{F}$. Pick $f \in \mathscr{F}$ and its $\delta$ neighbor $f_\delta \in \mathscr{F}_\delta$. Utilizing the deterministic relation $|\mathbf{zm}(X)| \leqslant |X| + |\mathbb{E}[X]|$ and using the fact that $f_\delta$ is in a $\delta$ neighborhood of $f$, we arrive at the following bounds

$$|\mathbf{zm}(f(\boldsymbol{x}) - f_\delta(\boldsymbol{x}))| \leqslant \delta(\|\boldsymbol{x}\|_{\ell_2} + \mathbb{E}[\|\boldsymbol{x}\|_{\ell_2}]). \tag{A.8}$$

$$|\mathbf{zm}(f(\boldsymbol{x}) + f_\delta(\boldsymbol{x}) - 2y)| \leqslant 2R(\|\boldsymbol{x}\|_{\ell_2} + \mathbb{E}[\|\boldsymbol{x}\|_{\ell_2}]) + 2K|\mathbf{zm}(y)|. \tag{A.9}$$

Next observe that, with probability at least $1 - 4n\exp(-p)$, all $\boldsymbol{x}_i, y_i$ obey $\|\boldsymbol{x}_i\|_{\ell_2} \leqslant \mathscr{O}(\sqrt{p})$, $|\mathbf{zm}(y_i)| \leqslant \mathscr{O}(K\sqrt{p})$. Combining this with (A.8), we conclude that for all $1 \leqslant i \leqslant n$

$$|\mathbf{zm}(f(\boldsymbol{x}_i) - y_i)^2 - \mathbf{zm}(f_\delta(\boldsymbol{x}_i) - y_i)^2| \leqslant \mathscr{O}(\bar{K}\delta p). \tag{A.10}$$

Expanding the square differences in the same way, an identical argument shows the following two deviation bounds:

$$|\mathbb{E}[\mathbf{zm}(f(\boldsymbol{x}_i)-y_i)^2-\mathbf{zm}(f_\delta(\boldsymbol{x}_i)-y_i)^2]| \leqslant \mathscr{O}(\bar{K}\delta p), \tag{A.11}$$

$$\frac{1}{n^2}|(\sum_{i=1}^n \mathbf{zm}(y_i-f(\boldsymbol{x}_i)))^2-(\sum_{i=1}^n \mathbf{zm}(y_i-f_\delta(\boldsymbol{x}_i)))^2| \leqslant \mathscr{O}(\bar{K}\delta p).$$

Combining these three inequalities ((A.10) and (A.11)) and substituting them in (A.6), we conclude that for all neighbors $f_\delta, f$,

$$|E(f)-E(f_\delta)| \leqslant \mathscr{O}(\bar{K}\delta p).$$

Next we set $\delta = c\varepsilon/(p\bar{K})$ for a sufficiently small constant $c > 0$, to find that with probability at least $1-\exp(-n)$, $\sup_{f\in\mathscr{F}}|E(f)| \leqslant \varepsilon$ holds as long as the number of samples $n$ obeys $n \geqslant \mathscr{O}(\max\{\varepsilon^{-1},\varepsilon^{-2}\}\bar{K}^2 s\log\frac{Cp\bar{K}}{c\varepsilon})$. We define $\mathscr{L}_{erm}(f) = \frac{1}{n}\sum_{i=1}^n(y_i-f(\boldsymbol{x}_i)-r_{\text{avg}}(f))^2$ and $\mathscr{L}_{pop}(f) = \mathbb{E}[\mathbf{zm}(f(\boldsymbol{x})-y)^2]$. We also denote the CERM minimizer $f_{erm} = \arg\min_{f\in\mathscr{F}}\mathscr{L}(f)$ and population minimizer $f_{pop} = \min_{f\in\mathscr{F}}\mathscr{L}_{pop}(f)$. Inequality (A.5) follows from the facts that we simultaneously have $|E(f_{erm})| \leqslant \mathscr{O}(\varepsilon)$ and $|E(f_{pop})| \leqslant \mathscr{O}(\varepsilon)$ which implies that

$$\mathscr{L}_{pop}(f_{erm}) \leqslant \mathscr{L}_{erm}(f_{erm})+\mathscr{O}(\varepsilon) \leqslant \mathscr{L}_{erm}(f_{pop})+\mathscr{O}(\varepsilon) \leqslant \mathscr{L}_{pop}(f_{pop})+\mathscr{O}(\varepsilon),$$

concluding the proof. □

### A.3 *Proof of Theorem A.1*

In this section, we will show how Theorem A.1 follows from Theorem A.2. To this aim, we need to show that $\mathscr{F}_{\hat{\boldsymbol{k}},B}$ has a small Lipscshitz covering number. We construct the following cover $\mathscr{F}'$ for the set $\mathscr{F}_{\hat{\boldsymbol{k}},B}$. Let $B' = B^{1/D}$. Pick a $\delta \leqslant B'$ $\ell_2$ cover $\mathscr{C}$ of the interval $[-B', B']$ which has size $2B'/\delta$. Let $\mathscr{C}_i$ be identical copies of $\mathscr{C}$. We set

$$\mathscr{F}' = \{f_{\text{CNN}}(\beta_\ell \hat{\boldsymbol{k}}^{(\ell)}) \mid \beta_\ell \in \mathscr{C}_\ell,\ 1 \leqslant \ell \leqslant D\}.$$

In words, we construct CNNs by picking numbers from the cartesian product $\mathscr{C}_1 \times \ldots \mathscr{C}_D$ and scaling $\{\hat{\boldsymbol{k}}^{(\ell)}\}_{\ell=1}^D$ with them. We now argue that $\mathscr{F}'$ provides a cover of $\mathscr{F}$. Given $f \in \mathscr{F}$ with scalings $\beta_\ell$, there exists $f' \in \mathscr{F}'$ which uses scalings $\beta'_\ell$ such that $|\beta_\ell - \beta'_\ell| \leqslant \delta$. Now, let $f_\ell$ be the function with scalings $\beta'_i$ until $i = \ell$ and $\beta_i$ for $i > \ell$. Note that $f_0 = f$, $f_D = f'$. With this, we write

$$\|f-f'\|_{L_\infty} \leqslant \sum_{i=1}^D \|f_{i+1}-f_i\|_{L_\infty}.$$

Observe that $f_{i-1}$ and $f_i$ have equal layers except the $i$th layer. Let $g_1$ be the function of the first $i-1$ layers and $g_2$ be the function of layers $i+1$ to $D$. We have that $f_{i+1}(\boldsymbol{x})-f_i(\boldsymbol{x}) = g_2(\phi(\boldsymbol{K}_i(g_1(\boldsymbol{x}))))-g_2(\phi(\boldsymbol{K}'_i(g_1(\boldsymbol{x}))))$ where $\boldsymbol{K}_i, \boldsymbol{K}'_i$ differ in the $i$th layer kernels of $f$ and $f'$ created from $\beta_i\hat{\boldsymbol{k}}^{(i)}$ and $\beta'_i\hat{\boldsymbol{k}}^{(i)}$ respectively. Also, observe that $g_1$ is $B'^{i-1}$ Lipschitz and $g_2(\phi(\cdot))$ is $B'^{D-i}$ Lipschitz functions. Hence,

$$|g_2(\phi(\boldsymbol{K}_i(g_1(\boldsymbol{x}))))-g_2(\phi(\boldsymbol{K}'_i(g_1(\boldsymbol{x}))))| \leqslant B'^{D-i}|\boldsymbol{K}_i(g_1(\boldsymbol{x}))-\boldsymbol{K}'_i(g_1(\boldsymbol{x}))| \leqslant B'^{D-i}\delta B'^{i-1} \leqslant \delta B'^{D-1}. \tag{A.12}$$

Summing over all $i$, this implies that $\|f - f'\|_{L_\infty} \leqslant D\delta B'^{D-1}$. Recalling $|\mathscr{F}'| \leqslant (2B'/\delta)^D$ and setting $\delta = \varepsilon/(DB'^{D-1})$, the $\varepsilon$ covering number of $\mathscr{F}_{\hat{\boldsymbol{k}},B}$ is $N_\varepsilon \leqslant (2DB'^D/\varepsilon)^D = (2DB/\varepsilon)^D$ which implies $\log N_\varepsilon = D\log(\frac{2DB}{\varepsilon})$. Now, since all kernels have Euclidean norm bounded by $B'$, we have $\|f_{\mathrm{CNN}}()\|_{L_\infty} \leqslant B$ and $\|f\|_{L_\infty} \leqslant B$ for all $f \in \mathscr{F}$. This also implies $\|\mathbf{zm}(f_{\mathrm{CNN}}(\boldsymbol{x}))\|_{\psi_2} = \mathscr{O}(B)$. Hence, we can apply Theorem A.2 to conclude the proof of Theorem A.1.