Research in
the Mathematical Sciences

RESEARCH

Approximation properties of deep ReLU CNNs

Juncai He[1*], Lin Li[2] and Jinchao Xu[3]

*Correspondence:
jhe@utexas.edu
[1]Department of Mathematics,
The University of Texas at Austin,
Austin, TX 78712, USA
Full list of author information is
available at the end of the article

Abstract

This paper focuses on establishing L^2 approximation properties for deep ReLU convolutional neural networks (CNNs) in two-dimensional space. The analysis is based on a decomposition theorem for convolutional kernels with a large spatial size and multi-channels. Given the decomposition result, the property of the ReLU activation function, and a specific structure for channels, a universal approximation theorem of deep ReLU CNNs with classic structure is obtained by showing its connection with one-hidden-layer ReLU neural networks (NNs). Furthermore, approximation properties are obtained for one version of neural networks with ResNet, pre-act ResNet, and MgNet architecture based on connections between these networks.

Keywords: Convolutional neural networks, Approximation properties, ResNet and MgNet

Mathematics Subject Classification: 41A30, 68T07, 65D40

1 Introduction

The purpose of this paper is to study the approximation properties of deep convolutional neural networks, including classic CNNs [22,25], ResNet [15], pre-act ResNet [16], and MgNet [13]. CNN is a very efficient deep learning model [8,24], which has been widely used in image processing, computer vision, reinforcement learning, and also scientific computing [10,19]. However, there is still very little mathematical analysis of CNNs and, therefore, limited understanding of them, especially for the approximation property of CNNs, which plays a functional role in their interpretation and development [21,27].

In the last three decades, researchers have produced a large number of studies on the approximation and representation properties of fully connected neural networks with a single hidden layer [2,4,5,17,20,26,37,40,43] and deep neural networks (DNNs) with more than one hidden layer [1,9,11,12,28,29,32,35,36,42,44]. To our knowledge, however, the literature included very few studies on the approximation property of CNNs [3,23,31,34,45,46]. In [3], the authors consider a type of ReLU CNN with one-dimensional (1D) input that is constituted by a sequence of convolution layers and a fully connected output layer. By showing that the identity operator can be realized by an underlying sequence of convolutional layers, they obtain the approximation property of the CNN directly from the fully connected layer. In their analysis, the underlying convolutional layers do not contribute anything to the approximation power of the overall CNN. Approximation properties with more standard CNN architecture have been studied in [45,46] in relation to the kernel decomposition for 1D convolutional operation with periodic padding. This type of result, however, cannot be extended to CNNs with two-dimensional (2D) inputs, because essentially it is the polynomial decomposition theory [6] for 1D. In [45,46], the authors study a standard 1D ReLU CNN architecture consisting of a sequence of convolution layers and a linear layer and obtain approximation properties by showing that any fully connected layer can be decomposed as a sequence of convolution layers with the ReLU activation function. In [34], the authors extend the analysis in [45,46] to 2D ReLU CNNs with periodic padding for a very special function class. This class is in the form of $f(X) = [F(X)]_{1,1} + b$ in which $F : \mathbb{R}^{d\times d} \mapsto \mathbb{R}^{d\times d}$ satisfies the following translation invariant property:

$$F(S_{st}(X)) = S_{st}(F(X)), \quad \forall X \in \mathbb{R}^{d\times d}, \tag{1.1}$$

where $S_{st} : \mathbb{R}^{d\times d} \mapsto \mathbb{R}^{d\times d}$ is considered the translation operator defined as $[S_{st}(X)]_{i,j} = [X]_{i-s,j-t}$ for $1 \leq s, t \leq d$ with periodic padding. Here, $[Y]$ means taking the element of the tensor Y. A generalized study of this function class and its application in approximation properties of CNNs can be found in [23]. In [31], the authors study the approximation properties of ResNet-type CNNs on 1D for the special function class that can be approximated by sparse NNs.

First, we show a pure algebraic decomposition theorem, which plays a critical role in establishing the approximation theorem of deep ReLU CNNs, for 2D convolutional kernels with multi-channel and constant or periodic padding. The core idea in establishing such a decomposition result is to introduce channels, whereas the decomposition theorem in [3,45,46] incorporates only one channel. By applying a similar argument in [46], we then establish a connection between one-hidden-layer ReLU NNs and deep ReLU CNNs without pooling layers. According to this connection, we prove the approximation theorem of classic deep ReLU CNNs, which shows that this kind of CNN can provide the same asymptotic approximation rate as one-hidden-layer ReLU NNs. Moreover, we obtain approximation results for ResNet and pre-act ResNet CNNs by studying connections between classic deep ReLU CNNs and CNNs with ResNet or pre-act ResNet architecture. Finally, we establish the approximation property of one version of MgNet [13] based on its connection with pre-act ResNet.

The paper is organized as follows. In Sect. 2, we introduce the 2D convolutional operation with multi-channel and paddings and then prove the decomposition theorem for large convolutional kernels. In Sect. 3, we show the approximation results for functions represented by classic CNNs without pooling operators. In Sect. 5, we provide concluding remarks.

2 Decomposition theorem of large convolutional kernels in CNNs

In this section, we introduce the decomposition theorem for standard two-dimensional convolutional kernels with large spatial size.

First, let us follow the setup for the dimensions of the tensors in PyTorch [33] to denote the data with c channels as

$$X \in \mathbb{R}^{c\times d\times d} \tag{2.1}$$

with elements $[X]_{p,m,n}$ for $p = 1 : c$ and $m, n = 1 : d$. For the convolutional kernel with input channel c, output channel C, and spatial size $(2k + 1) \times (2k + 1)$, we have

$$K \in \mathbb{R}^{c\times C\times(2k+1)\times(2k+1)} \tag{2.2}$$

with elements $[K]_{p,q,s,t}$ for $p = 1 : c$, $q = 1 : C$, and $s, t = -k : k$. Then, the standard multi-channel convolution operation in typical 2D CNNs [8] with constant or periodic padding is defined as $K * X \in \mathbb{R}^{C\times d\times d}$ where

$$[K * X]_{q,m,n} = \sum_{p=1}^{c} \sum_{s,t=-k}^{k} [K]_{p,q,s,t}[X]_{p,m+s,n+t} \tag{2.3}$$

for $q = 1 : C$ and $m, n = 1 : d$. If the index $m + s$ or $n + t$ exceeds the range $1 : d$ in (2.3), we denote the constant padding and the periodic padding as follows:

Constant padding:

$$[X]_{p,m+s,n+t} = a, \tag{2.4}$$

where $a \in \mathbb{R}$ is an arbitrary constant and $m + s \notin 1 : d$ or $n + t \notin 1 : d$;

Periodic padding:

$$[X]_{p,m+s,n+t} = [X]_{p,k,l}, \tag{2.5}$$

where $1 \leq k, l \leq d$, $k \equiv m + s \pmod d$, and $l \equiv n + t \pmod d$.

The convolution with constant or periodic padding defined in (2.3), referred to as convolution with stride one [8] with padding, is the most commonly used convolutional operation in practical CNNs [15,16,18,41]. An important feature of this convolution is that the spatial dimensions of its inputs do not change in the presence of paddings.

Remark 1 For simplicity, we assume the index of the convolution kernel $K \in \mathbb{R}^{(2k+1)\times(2k+1)}$ starts from $-k$ and ends at k, whereas the index of the data or tensor after convolution starts from 1. In addition, we stress that the convolution operation defined above follows neither the commutative law nor the associative law. Thus, we mean

$$K_2 * K_1 * X := K_2 * (K_1 * X) \tag{2.6}$$

by default.

Our study begins with the observation that a 5 × 5 kernel can be represented by the combination of two composed 3 × 3 kernels.

Lemma 1 *Let $K \in \mathbb{R}^{5\times 5}$ and $d > 2$, then there exist $P_{i,j}, S_{i,j} \in \mathbb{R}^{3\times 3}$ for $i, j = -1, 0, 1$ such that*

$$K * X = \sum_{i,j=-1,0,1} P_{i,j} * S_{i,j} * X, \quad \forall X \in \mathbb{R}^{d\times d}, \tag{2.7}$$

where $*$ means the standard convolution with one channel and constant or periodic padding as in (2.3).

Proof Here, we present a constructive proof by taking $S_{i,j}$ as

$$\left[S_{i,j}\right]_{s,t} = \begin{cases} 1 & s = i \text{ and } t = j, \\ 0 & \text{others}, \end{cases} \tag{2.8}$$

i.e.,

$$\begin{aligned}
&S_{-1,-1} = \begin{pmatrix} 1&0&0\\0&0&0\\0&0&0 \end{pmatrix}, \quad S_{-1,0} = \begin{pmatrix} 0&1&0\\0&0&0\\0&0&0 \end{pmatrix}, \quad S_{-1,1} = \begin{pmatrix} 0&0&1\\0&0&0\\0&0&0 \end{pmatrix},\\
&S_{0,-1} = \begin{pmatrix} 0&0&0\\1&0&0\\0&0&0 \end{pmatrix}, \quad S_{0,0} = \begin{pmatrix} 0&0&0\\0&1&0\\0&0&0 \end{pmatrix}, \quad S_{0,1} = \begin{pmatrix} 0&0&0\\0&0&1\\0&0&0 \end{pmatrix},\\
&S_{1,-1} = \begin{pmatrix} 0&0&0\\0&0&0\\1&0&0 \end{pmatrix}, \quad S_{1,0} = \begin{pmatrix} 0&0&0\\0&0&0\\0&1&0 \end{pmatrix}, \quad S_{1,1} = \begin{pmatrix} 0&0&0\\0&0&0\\0&0&1 \end{pmatrix},
\end{aligned} \tag{2.9}$$

and $P_{i,j}$ as

$$\begin{aligned}
&P_{-1,-1} = \begin{pmatrix} K_{-2,-2}&0&0\\0&0&0\\0&0&0 \end{pmatrix}, \quad P_{-1,0} = \begin{pmatrix} K_{-2,-1}&K_{-2,0}&K_{-2,1}\\0&0&0\\0&0&0 \end{pmatrix}, \quad P_{-1,1} = \begin{pmatrix} 0&0&K_{-2,2}\\0&0&0\\0&0&0 \end{pmatrix},\\
&P_{0,-1} = \begin{pmatrix} K_{-1,-2}&0&0\\K_{0,-2}&0&0\\K_{1,-2}&0&0 \end{pmatrix}, \quad P_{0,0} = \begin{pmatrix} K_{-1,-1}&K_{-1,0}&K_{-1,1}\\K_{0,-1}&K_{0,0}&K_{0,1}\\K_{1,-1}&K_{1,0}&K_{1,1} \end{pmatrix}, \quad P_{0,1} = \begin{pmatrix} 0&0&K_{-1,2}\\0&0&K_{0,2}\\0&0&K_{1,2} \end{pmatrix},\\
&P_{1,-1} = \begin{pmatrix} 0&0&0\\0&0&0\\K_{2,-2}&0&0 \end{pmatrix}, \quad P_{1,0} = \begin{pmatrix} 0&0&0\\0&0&0\\K_{2,-1}&K_{2,0}&K_{2,1} \end{pmatrix}, \quad P_{1,1} = \begin{pmatrix} 0&0&0\\0&0&0\\0&0&K_{2,2} \end{pmatrix}.
\end{aligned} \tag{2.10}$$

□

Remark 2 $S_{i,j}$ and $P_{i,j}$ can be collected separately to form two multi-channel convolution kernels. More precisely, we have

$$S = (S_{-1,-1}, S_{-1,0}, \cdots, S_{1,1}) \in \mathbb{R}^{1\times 9\times 3\times 3} \tag{2.11}$$

and

$$P = (P_{-1,-1}, P_{-1,0}, \cdots, P_{1,1})^T \in \mathbb{R}^{9\times 1\times 3\times 3}. \tag{2.12}$$

That is, the convolution operation defined in (2.7) can be written as

$$K * X = P * S * X. \tag{2.13}$$

Then, the most critical step is to extend Lemma 1 to a convolutional kernel $K \in \mathbb{R}^{(2k+1)\times(2k+1)}$ with large spatial size; i.e., k is large. Thus, we introduce the next decomposition for any $K \in \mathbb{R}^{(2k+1)\times(2k+1)}$ as

$$K = \sum_{i,j=-1,0,1} \widetilde{K}_{i,j}, \tag{2.14}$$

where

$$\begin{aligned}
\widetilde{K}_{-1,-1} &= \begin{pmatrix} K_{-k,-k} & 0 & \cdots \\ 0 & \ddots & \vdots \\ \vdots & \cdots & 0 \end{pmatrix} = \begin{pmatrix} P_{-1,-1} & & 0 & 0 \\ & & \vdots & \vdots \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & 0 \end{pmatrix} \in \mathbb{R}^{(2k+1)\times(2k+1)}, \\
\widetilde{K}_{-1,0} &= \begin{pmatrix} 0 & K_{-k,-k+1} & \cdots & K_{-k,k-1} & 0 \\ 0 & \ddots & 0 & \ddots & 0 \\ \vdots & \cdots & \vdots & \cdots & \vdots \end{pmatrix} = \begin{pmatrix} 0 & P_{-1,0} & 0 \\ \vdots & & \vdots \\ 0 & \cdots & 0 \end{pmatrix} \in \mathbb{R}^{(2k+1)\times(2k+1)}, \\
&\vdots \\
\widetilde{K}_{1,1} &= \begin{pmatrix} 0 & \cdots & \vdots \\ \vdots & \ddots & 0 \\ \cdots & 0 & K_{k,k} \end{pmatrix} = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \\ & & & P_{1,1} \\ 0 & 0 & & \end{pmatrix} \in \mathbb{R}^{(2k+1)\times(2k+1)},
\end{aligned} \tag{2.15}$$

and $P_{i,j} \in \mathbb{R}^{(2k-1)\times(2k-1)}$ with

$$P_{-1,-1} = \begin{pmatrix} K_{-k,-k} & 0 & \cdots \\ 0 & \ddots & \vdots \\ \vdots & \cdots & 0 \end{pmatrix}, \qquad P_{-1,0} = \begin{pmatrix} K_{-k,-k+1} & \cdots & K_{-k,k-1} \\ 0 & \ddots & 0 \\ \vdots & \cdots & \vdots \end{pmatrix},$$

$$P_{-1,1} = \begin{pmatrix} \cdots & 0 & K_{-k,k} \\ \vdots & \ddots & 0 \\ 0 & \cdots & \vdots \end{pmatrix},$$

$$P_{0,-1} = \begin{pmatrix} K_{-k+1,-k} & 0 & \cdots \\ \vdots & \ddots & \vdots \\ K_{k-1,-k} & 0 & \cdots \end{pmatrix}, \qquad P_{0,0} = \begin{pmatrix} K_{-k+1,-k+1} & \cdots & K_{-k+1,k-1} \\ \vdots & \ddots & \vdots \\ K_{k-1,-k+1} & \cdots & K_{k-1,k-1} \end{pmatrix},$$

$$P_{0,1} = \begin{pmatrix} \cdots & 0 & K_{-k+1,k} \\ \vdots & \ddots & \vdots \\ \cdots & 0 & K_{k-1,k} \end{pmatrix},$$

$$P_{1,-1} = \begin{pmatrix} \vdots & \cdots & 0 \\ 0 & \ddots & \vdots \\ K_{k,-k} & 0 & \cdots \end{pmatrix}, \qquad P_{1,0} = \begin{pmatrix} \vdots & \cdots & \vdots \\ 0 & \ddots & 0 \\ K_{k,-k+1} & \cdots & K_{k,k-1} \end{pmatrix},$$

$$P_{1,1} = \begin{pmatrix} 0 & \cdots & \vdots \\ \vdots & \ddots & 0 \\ \cdots & 0 & K_{k,k} \end{pmatrix}. \tag{2.16}$$

A more intuitive description of the previous decomposition is

$$K = \begin{pmatrix} \boxed{K_{-k,-k}} & \boxed{K_{-k,-k+1} \cdots K_{-k,k-1}} & \boxed{K_{-k,k}} \\ \boxed{\begin{matrix} K_{-k+1,-k} \\ \vdots \\ K_{k-1,-k} \end{matrix}} & \boxed{\begin{matrix} K_{-k+1,-k+1} & \cdots & K_{-k+1,k-1} \\ \vdots & \ddots & \vdots \\ K_{k-1,-k+1} & \cdots & K_{k-1,k-1} \end{matrix}} & \boxed{\begin{matrix} K_{-k+1,k} \\ \vdots \\ K_{k-1,k} \end{matrix}} \\ \boxed{K_{k,-k}} & \boxed{K_{k,-k+1} \cdots K_{k,k-1}} & \boxed{K_{k,k}} \end{pmatrix}. \tag{2.17}$$

Thus, we can regard $P_{i,j}$ in (2.16) as the generalization of $P_{i,j}$ in (2.10). Now, we present the main theorem for decomposing any large convolutional kernels $K \in \mathbb{R}^{(2k+1)\times(2k+1)}$.

Theorem 1 *Let $K \in \mathbb{R}^{(2k+1)\times(2k+1)}$ and $d > k$. Then, we can take $P_{i,j} \in \mathbb{R}^{(2k-1)\times(2k-1)}$ as in (2.16) and $S_{i,j} \in \mathbb{R}^{3\times 3}$ as in (2.8) for $i, j = -1, 0, 1$ such that*

$$K * X = \sum_{i,j=-1,0,1} P_{i,j} * S_{i,j} * X, \quad \forall X \in \mathbb{R}^{d\times d}, \tag{2.18}$$

where $*$ means the standard convolution with one channel and constant or periodic padding as in (2.3).

Proof Given the definition of $\widetilde{K}_{i,j}$ in (2.15), we need only verify that

$$\widetilde{K}_{i,j} * X = P_{i,j} * S_{i,j} * X \tag{2.19}$$

for any $i, j = -1, 0, 1$. For constant or periodic padding, we prove the above claim, respectively.

Periodic padding. For this case, we notice that

$$\left[S_{i,j} * X\right]_{m,n} = X_{m+i,n+j} \tag{2.20}$$

for any $i, j = -1, 0, 1$ and $1 \leq m, n \leq d$. Therefore, we have

$$\begin{aligned}\left[P_{i,j} * S_{i,j} * X\right]_{m,n} &= \sum_{p,q=-k+1,\cdots,k-1} \left[P_{i,j}\right]_{p,q} \left[S_{i,j} * X\right]_{m+p,n+q} \\ &= \sum_{p,q=-k+1,\cdots,k-1} \left[P_{i,j}\right]_{p,q} [X]_{m+p+i,n+q+j} \\ &= \sum_{\substack{\widetilde{p}=-k+1+i,\cdots k-1+i, \\ \widetilde{q}=-k+1+j,\cdots k-1+j}} \left[P_{i,j}\right]_{\widetilde{p}-i,\widetilde{q}-j} [X]_{m+\widetilde{p},n+\widetilde{q}}, \\ &= \sum_{\widetilde{p},\widetilde{q}=-k,\cdots,k} \left[\widetilde{K}_{i,j}\right]_{\widetilde{p},\widetilde{q}} [X]_{m+\widetilde{p},n+\widetilde{q}} \\ &= \left[\widetilde{K}_{i,j} * X\right]_{m,n}\end{aligned} \tag{2.21}$$

for any $i, j = -1, 0, 1$ and $1 \leq m, n \leq d$.

Constant padding. For this case, we split the proof into three cases according to different values of $|i| + |j|$.

1. $|i| + |j| = 0$, i.e., $i = j = 0$. Thus, for any $1 \leq m, n \leq d$, we have

$$\begin{aligned}\left[P_{0,0} * S_{0,0} * X\right]_{m,n} &= \sum_{p,q=-k+1,\cdots,k-1} \left[P_{0,0}\right]_{p,q} \left[S_{0,0} * X\right]_{m+p,n+q} \\ &= \sum_{p,q=-k+1,\cdots,k-1} [K]_{p,q} [X]_{m+p,n+q} \\ &= \left[\widetilde{K}_{0,0} * X\right]_{m,n}.\end{aligned} \tag{2.22}$$

2. $|i| + |j| = 2$, for example $(i, j) = (-1, -1)$ or $(1, -1)$. Without loss of generality, let us consider the example $(i, j) = (1, -1)$ first. Thus, we have

$$\begin{aligned}&\sum_{p,q=-k+1,\cdots,k-1} \left[P_{1,-1}\right]_{p,q} \left[S_{1,-1} * X\right]_{m+p,n+q} \\ &\quad = [K]_{k,-k} \left[S_{1,-1} * X\right]_{m+k-1,n-k+1}.\end{aligned} \tag{2.23}$$

As there is padding for $S_{1,-1} * X$ when we calculate $P_{1,-1} * S_{1,-1} * X$, it is necessary to compute $\left[S_{1,-1} * X\right]_{m+k-1,n-k+1}$ carefully. By definition, we first have $\left[S_{1,-1} * X\right]_{s,t}$ for $s, t = 1 : d$,

$$\left[S_{1,-1} * X\right]_{s,t} = \begin{cases} a, & \text{if } s = d \text{ or } t = 1, \\ [X]_{s+1,t-1}, & \text{others.} \end{cases} \tag{2.24}$$

We further mention that it is necessary to include padding in $S_{1,-1} * X$ in (2.23):

$$\left[S_{1,-1} * X\right]_{m+k-1,n-k+1} = \begin{cases} a, & \text{if } m \geq d - k + 2 \text{ or } n \leq k - 1, \\ \left[S_{1,-1} * X\right]_{m+k-1,n-k+1}, & \text{others.} \end{cases} \tag{2.25}$$

By combining the previous two equations and noticing that $k \geq 2$, we can obtain that

$$\begin{aligned}
&\left[S_{1,-1} * X\right]_{m+k-1,n-k+1} \\
&\quad = \begin{cases} a, & \text{if } m \geq d-k+1 \text{ or } n \leq k, \\ [X]_{m+k,n-k}, & \text{others}, \end{cases} \\
&\quad = [X]_{m+k,n-k}.
\end{aligned} \tag{2.26}$$

Therefore, we have

$$\begin{aligned}
\left[P_{1,-1} * S_{1,-1} * X\right]_{m,n} &= \sum_{p,q=-k+1,\cdots,k-1} \left[P_{1,-1}\right]_{p,q} \left[S_{1,-1} * X\right]_{m+p,n+q} \\
&= [K]_{k,-k} \left[S_{1,-1} * X\right]_{m+k-1,n-k+1} \\
&= [K]_{k,-k} [X]_{m+k,n-k} \\
&= \left[\widetilde{K}_{1,-1} * X\right]_{m,n}.
\end{aligned} \tag{2.27}$$

A similar derivation can be applied to the other three cases for $|i| + |j| = 2$.

3. $|i| + |j| = 1$, for example, $(i,j) = (-1,0)$ or $(0,1)$. Without loss of generality, let us consider the example $(i,j) = (1,-1)$. Thus, we have

$$\begin{aligned}
&\sum_{p,q=-k+1,\cdots,k-1} \left[P_{0,1}\right]_{p,q} \left[S_{0,1} * X\right]_{m+p,n+q} \\
&\quad = \sum_{p=-k+1,\cdots,k-1} [K]_{p,k} \left[S_{0,1} * X\right]_{m+p,n+k-1}.
\end{aligned} \tag{2.28}$$

First, let us take $p > 0$ and then compute $\left[S_{0,1} * X\right]_{m+p,n+k-1}$ in the same fashion. Thus, we have

$$\left[S_{0,1} * X\right]_{s,t} = \begin{cases} a, & \text{if } t = d, \\ [X]_{s,t+1}, & \text{others}, \end{cases} \tag{2.29}$$

and

$$\left[S_{0,1} * X\right]_{m+p,n+k-1} = \begin{cases} a, & \text{if } m \geq d-p+1 \text{ or } n \geq d-k+2, \\ \left[S_{1,-1} * X\right]_{m+p,n+k-1}, & \text{others}. \end{cases} \tag{2.30}$$

Furthermore, we can obtain that

$$\begin{aligned}
&\left[S_{0,1} * X\right]_{m+p,n+k-1} \\
&\quad = \begin{cases} a, & \text{if } m \geq d-p+1 \text{ or } n \geq d-k+1, \\ [X]_{m+p,n+k}, & \text{others}, \end{cases} \\
&\quad = [X]_{m+p,n+k}.
\end{aligned} \tag{2.31}$$

For $p < 0$, we can also go through the previous steps to reach the same conclusion. Thus, we have

$$\begin{aligned}\left[P_{0,1} * S_{0,1} * X\right]_{m,n} &= \sum_{p,q=-k+1,\cdots,k-1} \left[P_{0,1}\right]_{p,q} \left[S_{0,1} * X\right]_{m+p,n+q} \\ &= \sum_{p=-k+1,\cdots,k-1} [K]_{p,k} \left[S_{0,1} * X\right]_{m+p,n+k-1} \\ &= \sum_{p=-k+1,\cdots,k-1} [K]_{p,k} [X]_{m+p,n+k} \\ &= \left[\widetilde{K}_{0,1} * X\right]_{m,n}.\end{aligned} \tag{2.32}$$

A similar derivation can be applied to the other three cases for $|i| + |j| = 1$.

This completes the proof. □

According to the proof, the decomposition in (2.18) does not hold for arbitrary paddings such as reflection or replication padding [33], because equations (2.20), (2.26), and (2.31) cannot be true.

By applying the above theorem to decompose $P_{i,j}$ recursively, we have the following corollary.

Corollary 1 *Let $K \in \mathbb{R}^{(2k+1)\times(2k+1)}$ be a large kernel with one channel and $d > k$. Then, there exist $P_{(i_1,j_1),(i_2,j_2),\cdots,(i_{k-1},j_{k-1})} \in \mathbb{R}^{3\times3}$ and $S_{i_m,j_m} \in \mathbb{R}^{3\times3}$ for $i_m, j_m = -1, 0, 1$ and $m = 1 : k-1$ such that*

$$K * X = \sum_{i_{k-1},j_{k-1}} \cdots \sum_{i_1,j_1} P_{(i_1,j_1),\cdots,(i_{k-1},j_{k-1})} * S_{(i_{k-1},j_{k-1})} * \cdots * S_{(i_1,j_1)} * X \tag{2.33}$$

for any $X \in \mathbb{R}^{d\times d}$, where $*$ means the standard convolution with one channel as in (2.3).

Proof This can be proved by repeatedly applying Theorem 1 for $P_{i,j}$ in (2.18) until each $P_{(i_1,j_1),(i_2,j_2),\cdots,(i_{k-1},j_{k-1})}$ becomes a 3×3 kernel. □

As mentioned in Remark 2, we can collect all $P_{(i_1,j_1),(i_2,j_2),\cdots,(i_{k-1},j_{k-1})}$ into P as a single convolution kernel with multi-channels. Therefore, the output channel of P is 9^{k-1}, which will be huge if k is large. Thanks to the special pattern of zero in $P_{i,j}$ in (2.16), we have the following lemma to further reduce the number of nonzero output channels in P.

Lemma 2 *Let $K \in \mathbb{R}^{(2k+1)\times(2k+1)}$ and $d > k$. Then, there is an index set*

$$\boldsymbol{I}_{k-1} \subset \left\{((i_1,j_1),\cdots,(i_{k-1},j_{k-1})) \;\middle|\; i_m, j_m = \{-1,0,1\}, m = 1 : k-1\right\} \tag{2.34}$$

such that

$$K * X = \sum_{((i_1,j_1),\cdots,(i_{k-1},j_{k-1}))\in \boldsymbol{I}_{k-1}} P_{(i_1,j_1),\cdots,(i_{k-1},j_{k-1})} * S_{i_{k-1},j_{k-1}} * \cdots * S_{i_1,j_1} * X \tag{2.35}$$

for any $X \in \mathbb{R}^{d\times d}$, where $*$ means a standard convolution with one channel. Moreover, we have the cardinality of $\boldsymbol{I}_{k-1}$ as

$$\#\boldsymbol{I}_{k-1} = (2k-1)^2. \tag{2.36}$$

Proof This proof is based on the special distribution of zero for each $P_{i,j}$ in (2.16). Assume that we have applied Theorem 1 to $P_{i,j}$ for $n-1$-times with $n < k$ and obtained the following set of kernels:

$$\boldsymbol{P}_n := \left\{ P_{(i_1,j_1),\cdots,(i_n,j_n)} \;\middle|\; i_m, j_m = -1, 0, 1, m = 1:n \right\}. \tag{2.37}$$

It is easy to see that the cardinality of $\boldsymbol{P}_n$ is 9^n. Here, we prove that the number of nonzero items in $\boldsymbol{P}_n$ is bounded by $(2n+1)^2$. Because of the special form of $P_{i,j}$ in (2.16), we conclude that for nonzero $P_{(i_1,j_1),\cdots,(i_n,j_n)}$ there are only three types based on different zero-patterns.

1. Type 1: Nonzero items on the corner. For example, $P_{-1,-1}$ and $P_{-1,1}$ for $n = 1$, or $P_{(-1,-1),(-1,-1)}$ and $P_{(0,0),(1,-1)}$ for $n = 2$. We denote the number of elements with this type as C_n.
2. Type 2: Nonzero items on the boundary. For example, $P_{-1,0}$ and $P_{0,1}$ for $n = 1$, or $P_{(0,-1),(0,-1)}$ and $P_{(0,0),(1,0)}$ for $n = 2$. We denote the number of elements with this type as B_n.
3. Type 3: Full kernel. For example, $P_{0,0}$ for $n = 1$, or $P_{(0,0),(0,0)}$ for $n = 2$. A critical observation is that there is only one item with this form in $\boldsymbol{P}_n$ for any n, i.e., $P_{(0,0),\cdots,(0,0)} \in \boldsymbol{P}_n$.

The following rules describe the connections of the number of nonzero items between $\boldsymbol{P}_{n-1}$ and $\boldsymbol{P}_n$ when we apply Theorem 1 to $\boldsymbol{P}_{n-1}$ in order to obtain $\boldsymbol{P}_n$.

1. Type 1:

   $$C_n = C_{n-1} + 2B_{n-1} + 4, \tag{2.38}$$

   as each element in $\boldsymbol{P}_{n-1}$ with type 1 can make only one nonzero element in $\boldsymbol{P}_n$ with type 1, each element in $\boldsymbol{P}_{n-1}$ with type 2 can make two nonzero elements in $\boldsymbol{P}_n$ with type 1, and each element in $\boldsymbol{P}_{n-1}$ with type 3 can make four nonzero elements in $\boldsymbol{P}_n$ with type 1.
2. Type 2:

   $$B_n = B_{n-1} + 4, \tag{2.39}$$

   as each element in $\boldsymbol{P}_{n-1}$ with type 2 can make one nonzero element in $\boldsymbol{P}_n$ with type 2, each element in $\boldsymbol{P}_{n-1}$ with type 3 can make four nonzero elements in $\boldsymbol{P}_n$ with type 2, but each element in $\boldsymbol{P}_{n-1}$ with type 1 cannot make any nonzero element in $\boldsymbol{P}_n$ with type 2.
3. Type 3: There is only one nonzero element in $\boldsymbol{P}_n$. First, this nonzero item cannot be produced from elements in $\boldsymbol{P}_{n-1}$ with either type 1 or type 2. In addition, each element in $\boldsymbol{P}_{n-1}$ with type 3 can make only one nonzero element in $\boldsymbol{P}_n$ with type 3.

According to the decomposition in Theorem 1, we have

$$C_1 = B_1 = 4 \tag{2.40}$$

as the initial values for (2.38) and (2.39). Thus, we have

$$C_n = 4n^2 \quad \text{and} \quad B_n = 4n, \tag{2.41}$$

which means that the number of nonzero elements in $\boldsymbol{P}_n$ is

$$C_n + B_n + 1 = 4n^2 + 4n + 1 = (2n+1)^2. \tag{2.42}$$

Thus, the theorem is proved by taking $n = k-1$ and $\boldsymbol{I}_{k-1}$ as the index set of nonzero elements in $\boldsymbol{P}_{k-1}$. □

By representing the previous theorem in terms of convolution with multi-channels globally, we obtain the following theorem.

Theorem 2 *Let $K \in \mathbb{R}^{1\times M\times(2k+1)\times(2k+1)}$ and $d > k$. Then, there is a series of kernels $S^n \in \mathbb{R}^{c_{n-1}\times c_n\times 3\times 3}$ with multi-channels and $P \in \mathbb{R}^{(2k-1)^2\times M\times 3\times 3}$ such that*

$$K * X = P * S^{k-1} * S^{k-2} * \cdots * S^1 * X, \quad \forall X \in \mathbb{R}^{d\times d}, \tag{2.43}$$

where $c_n = (2n+1)^2$ for $n = 1 : k-1$ and $*$ means the standard convolution with multi-channels and padding as defined in (2.3).

Proof First, we follow the proof in Theorem 1 and notice that the index set $\boldsymbol{I}_n$ is independent from the kernel K and has this important feature:

$$((i_1,j_1),(i_2,j_2),\cdots,(i_n,j_n)) \in \boldsymbol{I}_n \Rightarrow ((i_1,j_1),(i_2,j_2),\cdots,(i_m,j_m)) \in \boldsymbol{I}_m, \tag{2.44}$$

if $m \le n$. Thus, we can define the following operator $\tau_n : \boldsymbol{I}_n \mapsto \boldsymbol{I}_{n-1}$ as

$$\tau_n\left(((i_1,j_1),(i_2,j_2),\cdots,(i_n,j_n))\right) = ((i_1,j_1),(i_2,j_2),\cdots,(i_{n-1},j_{n-1})). \tag{2.45}$$

Then, for each $\boldsymbol{I}_n$, we fix a bijection

$$\pi_n : \{1,2,\cdots,(2n+1)^n\} \mapsto \boldsymbol{I}_n \tag{2.46}$$

to give a unique position for each element in $\boldsymbol{I}_n$. For example, alphabetical order can be used. Thus, we construct $S^n \in \mathbb{R}^{c_{\ell-1}\times c_n\times 3\times 3}$ by taking

$$\left[S^n\right]_{p,q} = \begin{cases} S_{i_n,j_n}, & \text{if } \pi_n(q) = ((i_1,j_1),\cdots,(i_n,j_n)) \text{ and } \pi_{n-1}(p) = \tau_n(\pi_n(q)), \\ 0, & \text{others}, \end{cases} \tag{2.47}$$

for all $n = 1 : k-1$. Therefore, we can check that

$$\begin{aligned} \left[S^n * S^{n-1} * \cdots S^1 * X\right]_q &= \sum_{p=1}^{c_{n-1}} [S^n]_{p,q} * [S^{n-1} * \cdots S^1 * X]_p \\ &= S_{i_n,j_n} * [S^{n-1} * \cdots S^1 * X]_{\pi_{n-1}^{-1}(\tau_n(\pi_n(q)))} \\ &= S_{i_n,j_n} * S_{i_{n-1},j_{n-1}} * [S^{n-2} * \cdots S^1 * X]_{\pi_{n-2}^{-1}(\tau_{n-1}(\tau_n(\pi_n(q))))} \\ &= \cdots \\ &= S_{i_n,j_n} * S_{i_{n-1},j_{n-1}} * \cdots * S_{i_1,j_1} * X, \end{aligned} \tag{2.48}$$

where

$$((i_1, j_1), (i_2, j_2), \cdots, (i_n, j_n)) = \pi_n(q) \in \boldsymbol{I}_n \tag{2.49}$$

for all $1 \leq q \leq (2n+1)^2$. According to Theorem 1, for each channel $[K]_m \in \mathbb{R}^{(2k+1)\times(2k+1)}$ in $K \in \mathbb{R}^{1\times M\times(2k+1)\times(2k+1)}$, we have

$$[K]_m * X = \sum_{((i_1,j_1),\cdots,(i_{k-1},j_{k-1}))\in \boldsymbol{I}_{k-1}} P^m_{(i_1,j_1),\cdots,(i_{k-1},j_{k-1})} * S_{i_{k-1},j_{k-1}} * \cdots * S_{i_1,j_1} * X. \tag{2.50}$$

Finally, we finish the proof by constructing $P \in \mathbb{R}^{(2k-1)^2\times M\times 3\times 3}$ as

$$[P]_{p,m} = P^m_{\pi^{-1}_{k-1}(p)}, \tag{2.51}$$

where $P^m_{\pi^{-1}_{k-1}(p)}$ is defined in (2.50). □

3 Universal approximation theorem for classic CNNs

In this section, we show the universal approximation theorem for classic CNNs with 2D image inputs and standard multi-channel convolutions.

First, let us introduce CNN architecture with input data $x \in \mathbb{R}^{d\times d}$ and ReLU [30] activation function($\sigma(t) = \mathrm{ReLU}(t) := \max\{0, t\}$ for any $t \in \mathbb{R}$):

$$\begin{cases} f^\ell(x) & = \sigma(K^\ell * f^{\ell-1}(x) + b^\ell \mathbf{1}) \quad \ell = 1 : L, \\ f(x) & = a \cdot \mathcal{V}\left(f^L(x)\right), \end{cases} \tag{3.1}$$

where $f^0(x) = x \in \mathbb{R}^{d\times d}$, $K^\ell \in \mathbb{R}^{c_{\ell-1}\times c_\ell \times 3\times 3}$, $b^\ell \in \mathbb{R}^{c_\ell}$, $f^\ell \in \mathbb{R}^{c_\ell\times d\times d}$, $a \in \mathbb{R}^{c_L d^2}$, and $\mathcal{V}(f^L(x))$ denotes the vectorization of $f^L(x) \in \mathbb{R}^{c_L\times d\times d}$ by taking

$$\left[\mathcal{V}\left(f^L(x)\right)\right]_{(c-1)d^2+(s-1)d+t} = \left[f^L(x)\right]_{c,s,t} \tag{3.2}$$

for all $s, t = 1 : d$ and $c = 1 : c_L$. For simplicity, we extend the definition of $\mathcal{V}(\cdot)$ for the general tensor in $\mathbb{R}^{d\times d}$, $\mathbb{R}^{c_\ell\times d\times d}$, etc. Here, $K^\ell * f^{\ell-1}(x)$ follows the definition of convolution with multi-channel and constant or periodic padding as in (2.3). In addition, we consider the special form of bias in CNNs,

$$b^\ell \mathbf{1} := \left([b^\ell]_1 \mathbf{I}, [b^\ell]_2 \mathbf{I}, \cdots, [b^\ell]_{c_\ell} \mathbf{I}\right) \in \mathbb{R}^{c_\ell\times d\times d}, \tag{3.3}$$

where $\mathbf{I} \in \mathbb{R}^{d\times d}$ with $[\mathbf{I}]_{s,t} = 1$ for all $s, t = 1 : d$. Moreover, we notice that there is no pooling, subsampling, or coarsening operator (layer) to apply in the above CNN architecture. Furthermore, to investigate the approximation properties of CNNs on $\mathbb{R}^{d\times d}$, we consider $\mathbb{R}^{d\times d}$ as a d^2-dimensional vector space with Frobenius norm.

Before we prove the main approximation theorem, let us introduce the next two lemmas, which reveal the connection between deep ReLU CNNs and one-hidden-layer ReLU NNs.

Lemma 3 *For any $W \in \mathbb{R}^{N\times d^2}$ and $\alpha, \beta \in \mathbb{R}^N$, there is a convolutional kernel $K \in \mathbb{R}^{1\times N\times(2\lfloor d/2\rfloor+1)\times(2\lfloor d/2\rfloor+1)}$, bias $b \in \mathbb{R}^N$, and weight $a \in \mathbb{R}^{Nd^2}$ such that*

$$\alpha\cdot\sigma\left(W\mathcal{V}(x)+\beta\right)=a\cdot\mathcal{V}\left(\sigma\left(K*x+b\mathbf{1}\right)\right) \tag{3.4}$$

for any $x \in \mathbb{R}^{d\times d}$.

Proof For simplicity, let us first assume that d is odd. Then, we have $d = 2\lfloor d/2\rfloor + 1$ and

$$[K*x]_{n,\lceil d/2\rceil,\lceil d/2\rceil}=\sum_{s,t=-\lfloor d/2\rfloor}^{\lfloor d/2\rfloor}[K]_{n,s,t}[x]_{\lceil d/2\rceil+s,\lceil d/2\rceil+t}=\mathcal{V}([K]_n)\cdot\mathcal{V}(x). \tag{3.5}$$

Thus, this proof is completed by taking $b = \beta$,

$$\mathcal{V}([K]_{n,:,:})=[W]_{n,:}, \quad \text{and} \quad [a]_k=\begin{cases}[\alpha]_n, & \text{if } k=(n-1)d^2+\lceil d/2\rceil^2,\\ 0, & \text{others,}\end{cases} \tag{3.6}$$

for all $n = 1 : N$. If d is even, we have $2\lfloor d/2\rfloor + 1 = d + 1$. Thus, we can construct a and b as before and then take $\mathcal{V}([K]_{n,-d/2:d/2-1,-d/2:d/2-1}) = [W]_{n,:}$ and $[K]_{n,d/2,-d/2:d/2} = [K]_{n,-d/2:d/2,d/2} = 0$. Thus, we have

$$\begin{aligned}[K*x]_{n,d/2+1,d/2+1}&=\sum_{s,t=-d/2}^{d/2}[K]_{n,s,t}[x]_{d/2+1+s,d/2+1+t}\\&=\mathcal{V}([K]_{n,-d/2:d/2-1,-d/2:d/2-1})\cdot\mathcal{V}(x),\end{aligned} \tag{3.7}$$

which finishes the proof. □

Basically, the above lemma shows that a ReLU NN function with one hidden layer can be represented by a one-layer CNN with a large kernel.

Lemma 4 *For any bounded set $\Omega \subset \mathbb{R}^{d\times d}$, kernel $K \in \mathbb{R}^{1\times N\times(2\lfloor d/2\rfloor+1)\times(2\lfloor d/2\rfloor+1)}$, and bias vector $b \in \mathbb{R}^N$, there is a series of kernels $K^\ell \in \mathbb{R}^{c_{\ell-1}\times c_\ell\times 3\times 3}$ and bias vectors $b^\ell \in \mathbb{R}^{c_\ell}$ such that*

$$[K*x+b\mathbf{1}]_{n,\lceil d/2\rceil,\lceil d/2\rceil}=\left[K^{\lfloor d/2\rfloor}*f^{\lfloor d/2\rfloor-1}(x)+b^{\lfloor d/2\rfloor}\mathbf{1}\right]_{n,\lceil d/2\rceil,\lceil d/2\rceil}, \ \forall x\in\Omega, \tag{3.8}$$

where $f^0(x) = x$, $c_\ell = (2\ell + 1)^2$ for $\ell = 1 : \lfloor d/2\rfloor - 1$, $c_{\lfloor d/2\rfloor} = N$, and

$$f^\ell(x)=\sigma\left(K^\ell*f^{\ell-1}(x)+b^\ell\mathbf{1}\right). \tag{3.9}$$

Proof According to Theorem 2, we know there is $P \in \mathbb{R}^{(2\lfloor d/2\rfloor-1)^2\times N\times 3\times 3}$ and $S^\ell \in \mathbb{R}^{c_{\ell-1}\times c_\ell\times 3\times 3}$ with $c_\ell = (2\ell + 1)^2$ for $\ell = 1 : \lfloor d/2\rfloor - 1$ such that

$$K*x=P*S^{\lfloor d/2\rfloor-1}*\cdots*S^1*x \tag{3.10}$$

for any $x \in \Omega$. Thus, we can take

$$K^\ell = S^\ell, \quad \ell = 1 : \lfloor d/2 \rfloor - 1 \quad \text{and} \quad K^{\lfloor d/2 \rfloor} = P. \tag{3.11}$$

Then, we can prove (3.8) by taking b^ℓ in an appropriate way. From $\ell = 1$ to $\lfloor d/2 \rfloor - 1$, we define b^ℓ consecutively as

$$[b^\ell]_q = \max_{1 \leq s,t \leq d} \sup_{x \in \Omega} \left| \left[K^\ell * f^{\ell-1}(x) \right]_{q,s,t} \right|, \quad q = 1 : c_\ell. \tag{3.12}$$

As $\Omega \subset \mathbb{R}^{d \times d}$ is bounded and $\sigma = \text{ReLU}$ is continuous, we know that

$$[b^\ell]_q < \infty, \quad \forall \ell = 1 : \lfloor d/2 \rfloor - 1, \quad q = 1 : c_\ell. \tag{3.13}$$

Therefore, we have

$$f^\ell(x) = \sigma\left(K^\ell * f^{\ell-1}(x) + b^\ell \mathbf{1} \right) = K^\ell * f^{\ell-1}(x) + b^\ell \mathbf{1} \tag{3.14}$$

because of the definition of $\sigma(x)$ and b^ℓ. Thus, we have

$$K^{\lfloor d/2 \rfloor} * f^{\lfloor d/2 \rfloor - 1}(x) = P * S^{\lfloor d/2 \rfloor - 1} * \cdots * S^1 * x + B = K * x + B, \tag{3.15}$$

where

$$B = \sum_{\ell=2}^{\lfloor d/2 \rfloor - 1} P * S^{\lfloor d/2 \rfloor - 1} * \cdots * S^\ell * (b^{\ell-1}\mathbf{1}) + P * (b^{\lfloor d/2 \rfloor - 1}\mathbf{1}) \in \mathbb{R}^{N \times d \times d}, \tag{3.16}$$

which is constant in respect to x. Finally, we take

$$[b^{\lfloor d/2 \rfloor}]_n = [b]_n - [B]_{n, \lceil d/2 \rceil, \lceil d/2 \rceil}, \tag{3.17}$$

which finishes the proof. □

Lemma 4 shows that a one-layer ReLU CNN with a large kernel can be represented by a deep ReLU CNN with multi-channel 3×3 kernels.

To obtain our final theorem (Theorem 4), let us first recall the following approximation result for one-hidden-layer ReLU NNs.

Theorem 3 ([2,39]) *Assume that $f : \Omega \subset \mathbb{R}^D \mapsto \mathbb{R}$ and that Ω is bounded. Then, there is a ReLU NN with one hidden layer $f_N(x) = \alpha \cdot \sigma(Wx + \beta)$ where $W \in \mathbb{R}^{N \times D}$ and $\alpha, \beta \in \mathbb{R}^N$, such that*

$$\|f - f_N\|_{L^2(\Omega)} \lesssim N^{-\frac{1}{2} - \frac{3}{2D}} \|f\|_{\mathscr{K}_1(\mathbb{D})}. \tag{3.18}$$

Here, $a \lesssim b$ means $a \leq Cb$ where C depends only on dimension D and domain Ω. In addition, $\|f\|_{\mathscr{K}_1(\mathbb{D})}$ is the norm defined by the gauge of $B_1(\mathbb{D})$,

$$\|f\|_{\mathscr{K}_1(\mathbb{D})} = \inf\{c > 0 \ : f \in cB_1(\mathbb{D})\}, \tag{3.19}$$

where $B_1(\mathbb{D})$ is given by

$$B_1(\mathbb{D}) = \overline{\left\{ \sum_{j=1}^{n} a_j h_j \ : \ n \in \mathbb{N}, h_j \in \mathbb{D}, \sum_{j=1}^{n} |a_j| \leq 1 \right\}}, \tag{3.20}$$

and

$$\mathbb{D} = \{\sigma(\omega \cdot x + b) \ : \ \omega \in \mathbb{R}^D, b \in \mathbb{R}\} \tag{3.21}$$

is the dictionary generated by the activation function $\sigma(t) = \mathrm{ReLU}(t)$. More details about the $\|f\|_{\mathscr{K}_1(\mathbb{D})}$ norm, its approximation properties, and its connections with what is known as Barron norm can be found in [7,38,39]. Generally, the underlying model for image classification is a piecewise constant function for which it is impossible to have a finite $\mathscr{K}_1(\mathbb{D})$ norm. However, the ReLU CNN functions that we discuss in this paper are the feature extraction parts of ReLU CNN models for image classification. Thus, $f(x)$ may have a finite $\mathscr{K}_1(\mathbb{D})$ norm as a feature extraction function not a classification model.

By combining Lemma 3, Lemma 4, and Theorem 3, we have the following approximation theorem of deep CNNs with multi-channel 3 × 3 kernels.

Theorem 4 *Let $\Omega \subset \mathbb{R}^{d\times d}$ be bounded. Assume that $f : \Omega \mapsto \mathbb{R}$ and $\|f\|_{\mathscr{K}_1(\mathbb{D})} < \infty$. Then, there is a CNN function $\widetilde{f} : R^{d\times d} \mapsto \mathbb{R}$ as defined in (3.1) with multi-channel 3×3 kernels, where*

$$\begin{aligned} &\textit{depth (number of convolutional layers):} && L = \lfloor d/2 \rfloor, \\ &\textit{width (number of channels at each layer):} && c_\ell = (2\ell+1)^2, \end{aligned} \tag{3.22}$$

for $\ell = 1 : L-1$ and $c_L = N$, such that

$$\|f - \widetilde{f}\|_{L^2(\Omega)} \lesssim N^{-\frac{1}{2}-\frac{3}{2d^2}} \|f\|_{\mathscr{K}_1(\mathbb{D})}. \tag{3.23}$$

Proof First, we assume that $f_N(\mathcal{V}(x))$ is the approximation of $f(x)$ using a one-hidden-layer ReLU NN, as shown in Theorem 3. According to Lemma 3 and Lemma 4, there is a CNN $\widetilde{f}(x)$ as defined in (3.1) with the hyperparameters listed above such that $\widetilde{f}(x) = f_N(\mathcal{V}(x))$ for any $x \in \Omega$. This completes the proof. □

Here, we notice that the total number of free parameters in $f_N(\mathcal{V}(x))$, as shown in Theorem 3, is $\mathcal{N}_F = N(d^2+2)$ if $x \in \mathbb{R}^{d\times d}$. We also note that the total number of free parameters of the CNN function $\widetilde{f}(x)$ as in (3.1) with hyperparameters in Theorem 4 is

$$\begin{aligned} \mathcal{N}_C = &\sum_{\ell=1}^{\lfloor d/2 \rfloor -1} \left(\underbrace{9(2\ell+1)^2(2\ell-1)^2}_{K^\ell} + \underbrace{(2\ell+1)^2}_{b^\ell} \right) \\ &+ \underbrace{N\left((2\lfloor d/2 \rfloor -1)^2 + 1\right)}_{K^{\lfloor d/2 \rfloor}\ \&\ b^{\lfloor d/2 \rfloor}} + \underbrace{Nd^2}_{a} \\ &\leq 2(d^5 + Nd^2). \end{aligned} \tag{3.24}$$

A comparison of $\mathcal{N}_F$ and $\mathcal{N}_C$ shows that the convolutional neural networks with hyperparameters (depth L and width c_ℓ) in Theorem 4 can achieve the same asymptotic approximation order of one-hidden-layer ReLU NNs as $\mathcal{O}\left(N^{-\frac{1}{2}-\frac{3}{2d^2}}\right)$. That is, there is an upper bound for the approximation error as $CN^{-\frac{1}{2}-\frac{3}{2d^2}}$ where C depends only on dimension d and domain Ω.

In addition, in Theorem 4, to achieve the approximation power for ReLU CNNs, it is necessary for the depth to exceed $d/2$ and for the number of layers in ℓth layer to be at least $(2\ell+1)^2$. However, it is by no means common for practical CNN models to meet both of these conditions. Here, we may interpret these conditions from other perspectives. For the depth (number of layers) of CNN models for image classification, we usually apply ResNet [15] CNNs with 18 or 34 layers for CIFAR-10 and CIFAR-100 with input images in $\mathbb{R}^{32\times 32}$. Furthermore, we prefer deeper ResNet CNNs, for example, ResNet [16] with 50, 101, or more than 1,000 layers for ImageNet, which has input images in $\mathbb{R}^{224\times 224}$. In all these examples, depth L is greater than $d/2$ for input images in $\mathbb{R}^{d\times d}$. For the width (number of channels) of CNN models for image classification, we notice that every practical CNN model increases the input channel to a relatively large number and retains this width for several layers. These two observations indicate that it is necessary to include more layers and channels in practical CNN models.

Moreover, Theorem 4 requires a huge number of channels in the last layer to achieve a small approximate error. However, the number of channels in the last layer in practical CNNs is not very large in general. For example, one may take 512, 1,024, or 2,048 channels in the output layer for CIFAR and ImageNet classification problems. To understand this, we recall that the target function in this approximation result is the feature extraction function not the piecewise constant classification function. Thus, the $\|f\|_{\mathscr{K}_1(\mathbb{D})}$ norm may be very small such that a relatively small N may be enough to achieve sufficient approximation power. This implies that the feature extraction functions in image classification may lie in a special function class which is much smaller than $\mathscr{K}_1(\mathbb{D})$.

Furthermore, Theorem 4 does not tell us why CNNs are much better than DNNs for image classification in terms of approximation power. However, Theorem 4 does indicate that CNNs with certain structures are no worse than DNNs in terms of approximation. This is important when CNNs are applied in fields in which approximation accuracy is a critical metric, such as numerical solutions of PDEs [19]. On the other hand, Theorem 4 shows that deep ReLU CNNs with certain structures can represent ReLU NNs with one hidden layer. This implies that the function class of ReLU CNNs is much larger than the function class of one-hidden-layer ReLU NNs. More precisely, the function class of ReLU CNNs may include or can efficiently approximate some very special functions that cannot be approximated directly using ReLU DNNs.

4 Approximation properties of ResNet and MgNet

In this section, we show the approximation properties for one version of ResNet [15], pre-act ResNet [16], and MgNet [13].

Approximation properties of ResNet and pre-act ResNet. First, let us introduce some mathematical formulas to define these two network functions:

ResNet:

$$\begin{cases} f^\ell(x) = \sigma\left(R^\ell * f^{\ell-1}(x) + B^\ell * \sigma\left(A^\ell * f^{\ell-1}(x) + a^\ell \mathbf{1}\right) + b^\ell \mathbf{1}\right), \\ f(x) = a \cdot \mathcal{V}\left(f^L(x)\right), \end{cases} \tag{4.1}$$

for $\ell = 1 : L$, where $f^0(x) = x \in \mathbb{R}^{d\times d}$, $A^\ell \in \mathbb{R}^{c_{\ell-1}\times C_\ell \times 3\times 3}$, $B^\ell \in \mathbb{R}^{C_\ell \times c_\ell \times 3\times 3}$, $R^\ell \in \mathbb{R}^{c_{\ell-1}\times c_\ell\times 1\times 1}$, $a^\ell \in \mathbb{R}^{C_\ell}$, b^{c_ℓ}, and $a \in \mathbb{R}^{c_L d^2}$.

Pre-act ResNet:

$$\begin{cases} f^\ell(x) = R^\ell * f^{\ell-1}(x) + \sigma\left(B^\ell * \sigma\left(A^\ell * f^{\ell-1}(x) + a^\ell \mathbf{1}\right) + b^\ell \mathbf{1}\right), \\ f(x) = a \cdot \mathcal{V}\left(f^L(x)\right), \end{cases} \tag{4.2}$$

for $\ell = 1 : L$, where $f^0(x) = x \in \mathbb{R}^{d\times d}$, $A^\ell \in \mathbb{R}^{c_{\ell-1}\times C_\ell \times 3\times 3}$, $B^\ell \in \mathbb{R}^{C_\ell \times c_\ell \times 3\times 3}$, $R^\ell \in \mathbb{R}^{c_{\ell-1}\times c_\ell\times 1\times 1}$, $a^\ell \in \mathbb{R}^{C_\ell}$, b^{c_ℓ}, and $a \in \mathbb{R}^{c_L d^2}$.

Each iteration from $f^{\ell-1}(x)$ to $f^\ell(x)$ is called a basic block of ResNet or pre-act ResNet. Note here that we add an extra 1×1 kernel in front of $f^{\ell-1}(x)$ in each block, introduced in [15] originally, to adjust to the change of channels from $f^{\ell-1}(x)$ to $f^\ell(x)$. In addition, we notice that ResNet and pre-act ResNet can degenerate into a classical CNN as in (3.1), if we take $R^\ell = 0$. In this sense, ResNet and pre-act ResNet have approximation properties if we assume that R^ℓ is not given a priori but set as a trainable parameter. However, a key reason for the success of ResNet is that we set R^ℓ a priori, especially when taking R^ℓ as the identity operator when $f^{\ell-1}(x)$ and $f^\ell(x)$ have the same number of channels [15,16]. Thus, we consider ResNet and pre-act ResNet networks with an arbitrarily given value for R^ℓ.

Theorem 5 *Let $\Omega \subset \mathbb{R}^{d\times d}$ be bounded, and assume that $f : \Omega \mapsto \mathbb{R}$ with $\|f\|_{\mathscr{K}_1(\mathbb{D})} < \infty$. For any given $R^\ell \in \mathbb{R}^{c_{\ell-1}\times c_\ell \times 1\times 1}$, there is a ResNet network $\widetilde{f}(x)$ as in (4.1), where the hyperparameters satisfy*

$$\begin{aligned} \textit{depth (number of blocks):} \quad & L = \lfloor \lfloor d/2 \rfloor /2 \rfloor, \\ \textit{width (number of channels):} \quad & c_\ell = (4\ell+1)^2, \;\; C_\ell = 2(4\ell-1)^2 \end{aligned} \tag{4.3}$$

for $\ell = 1 : L-1$, $C_L = 2(4L-1)^2$, and $c_L = N$, such that

$$\|f - \widetilde{f}\|_{L^2(\Omega)} \lesssim N^{-\frac{1}{2}-\frac{3}{2d^2}} \|f\|_{\mathscr{K}_1(\mathbb{D})}. \tag{4.4}$$

Proof Without loss of generality, let us assume that $\frac{d}{4}$ is an integer. If not, some zero boundary layers can be added to enlarge the dimension of the original data to satisfy this condition. First, we assume a CNN function $\widehat{f}(x) = f_N(\mathcal{V}(x)) = \alpha \cdot \sigma\left(W\mathcal{V}(x) + \beta\right)$ defined in (3.1), which approximates $f(x)$, as described in Theorem 4:

$$\|f - \widehat{f}\|_{L^2(\Omega)} \lesssim N^{-\frac{1}{2}-\frac{3}{2d^2}} \|f\|_{\mathscr{K}_1(\mathbb{D})}.$$

Then, we can construct $\widetilde{f}(x)$ using kernels K^ℓ for $\ell = 1 : d/2$, $\widehat{b}^{d/2}$ and $\widehat{a}$ in $\widehat{f}(x)$. We define

$$A^\ell = \begin{pmatrix} K^{2\ell-1} \\ I \\ 0 \end{pmatrix}, \quad [a^\ell]_q = \max_{m,n} \sup_{x\in\Omega} \left| \left[A^\ell * f^{\ell-1}(x) \right]_{q,m,n} \right|, \quad \ell = 1 : L, \tag{4.5}$$

where I is the identity kernel and 0 is used to pad the output channel to be C_ℓ. That is,

$$\sigma\left(A^\ell * f^{\ell-1}(x) + a^\ell \mathbf{1}\right) = \begin{pmatrix} K^{2\ell-1} * f^{\ell-1} \\ f^{\ell-1} \\ 0 \end{pmatrix} + a^\ell \mathbf{1}. \tag{4.6}$$

Moreover, we define

$$B^\ell = \left(K^{2\ell}, -R^\ell, 0\right), \quad \ell = 1 : L \tag{4.7}$$

and

$$[b^\ell]_q = \max_{1\le s,t\le d} \sup_{x\in\Omega} \left| \left[R^\ell * f^{\ell-1}(x) + B^\ell * \left(A^\ell * f^{\ell-1}(x) + a^\ell \mathbf{1}\right)\right]_{q,s,t} \right| \tag{4.8}$$

for $\ell = 1 : L-1$. Given the definition of b^ℓ, it follows that $\left[R^\ell * f^{\ell-1}(x) + B^\ell * \sigma\left(A^\ell * f^{\ell-1}(x) + a^\ell \mathbf{1}\right) + b^\ell \mathbf{1}\right]_{q,s,t} \ge 0$. Given the definition of $\sigma(t) = \mathrm{ReLU(t)} := \max\{0, \mathrm{t}\}$, we have

$$\begin{aligned} f^\ell(x) &= \sigma\left(R^\ell * f^{\ell-1}(x) + B^\ell * \sigma\left(A^\ell * f^{\ell-1}(x) + a^\ell \mathbf{1}\right) + b^\ell \mathbf{1}\right) \\ &= R^\ell * f^{\ell-1}(x) + B^\ell * \left(A^\ell * f^{\ell-1}(x) + a^\ell \mathbf{1}\right) + b^\ell \mathbf{1} \\ &= R^\ell * f^{\ell-1}(x) + K^{2\ell} * K^{2\ell-1} * f^{\ell-1}(x) - R^\ell * f^{\ell-1}(x) + B^\ell * a^\ell \mathbf{1} + b^\ell \mathbf{1} \\ &= K^{2\ell} * K^{2\ell-1} * f^{\ell-1}(x) + B^\ell * a^\ell \mathbf{1} + b^\ell \mathbf{1} \end{aligned} \tag{4.9}$$

for $\ell = 1 : L-1$. By taking $\ell = L$, we have

$$\begin{aligned} f^L(x) &= R^L * f^{L-1}(x) + B^L * \sigma\left(A^L * f^{L-1}(x) + a^L \mathbf{1}\right) + b^L \mathbf{1} \\ &= K^{2L} * K^{2L-1} * f^{L-1}(x) + K^{2L} * a^L \mathbf{1} + b^L \mathbf{1} \\ &= K^{d/2} * K^{d/2-1} * \cdots * K^1 * x + B, \end{aligned} \tag{4.10}$$

where B is a constant tensor, which is similar to the case presented in Lemma 4. This suggests b^L defined as

$$[b^L]_n = [\beta]_n - [B]_{n,d/2,d/2}, \tag{4.11}$$

where $\beta \in \mathbb{R}^N$ proceeds from $\widehat{f}(x) = f_N(\mathcal{V}(x)) = \alpha \cdot \sigma\left(W\mathcal{V}(x) + \beta\right)$ for any $x \in \Omega$. Thus, it follows that

$$f^L(x) = \widehat{f}^L(x), \tag{4.12}$$

and we complete the proof by taking $a = \widehat{a}$. □

Theorem 6 *Let $\Omega \subset \mathbb{R}^{d\times d}$ be bounded, and assume that $f : \Omega \mapsto \mathbb{R}$ with $\|f\|_{\mathscr{K}_1(\mathbb{D})}$. For any given $R^\ell \in \mathbb{R}^{c_{\ell-1}\times c_\ell \times 1\times 1}$, there is a pre-act ResNet network $\widetilde{f}(x)$ as in (4.2), where the hyperparameters satisfy*

$$\begin{aligned} &\textit{depth (number of blocks):} && L = \lfloor \lfloor d/2 \rfloor /2 \rfloor, \\ &\textit{width (number of channels):} && c_\ell = (4\ell+1)^2,\ C_\ell = 2(4\ell-1)^2 \end{aligned} \tag{4.13}$$

for $\ell = 1 : L-1$, $C_L = 2(4L-1)^2$ and $c_L = N+2$, such that

$$\|f - \widetilde{f}\|_{L^2(\Omega)} \lesssim N^{-\frac{1}{2}-\frac{3}{2d^2}} \|f\|_{\mathscr{K}_1(\mathbb{D})}. \tag{4.14}$$

Proof We still assume that $\frac{d}{4}$ is an integer. Then, we construct pre-act ResNet as in (4.2) with a similar structure as that described in Theorem 5. According to Lemma 3 and Lemma 4, we notice that kernels K^ℓ for $\ell = 1 : \lfloor d/2 \rfloor - 1$ as described in Theorem 4 are independent from $f(x)$. More precisely, these kernels can be defined by using S^ℓ presented in Theorem 2. Thus, we take

$$A^\ell = \begin{pmatrix} K^{2\ell-1} \\ I \\ 0 \end{pmatrix}, \quad [a^\ell]_q = \max_{m,n} \sup_{x\in\Omega} \left| \left[A^\ell * f^{\ell-1}(x) \right]_{q,m,n} \right| \tag{4.15}$$

for $\ell = 1 : L$ where I is the identity kernel and 0 is used to pad the output channel to be C_ℓ. That is,

$$\sigma\left(A^\ell * f^{\ell-1}(x) + a^\ell \mathbf{1} \right) = \begin{pmatrix} K^{2\ell-1} * f^{\ell-1} \\ f^{\ell-1} \\ 0 \end{pmatrix} + a^\ell \mathbf{1}. \tag{4.16}$$

Moreover, we define

$$B^\ell = \left(K^{2\ell}, -R^\ell, 0 \right), \quad \ell = 1 : L-1 \tag{4.17}$$

and

$$[b^\ell]_q = \max_{1\le s,t\le d} \sup_{x\in\Omega} \left| \left[B^\ell * \left(A^\ell * f^{\ell-1}(x) + a^\ell \mathbf{1} \right) \right]_{q,s,t} \right|, \quad \ell = 1 : L-1. \tag{4.18}$$

This means that

$$\begin{aligned} f^\ell(x) &= R^\ell * f^{\ell-1}(x) + \sigma\left(B^\ell * \sigma\left(A^\ell * f^{\ell-1}(x) + a^\ell \mathbf{1} \right) + b^\ell \mathbf{1} \right) \\ &= R^\ell * f^{\ell-1}(x) + B^\ell * \left(A^\ell * f^{\ell-1}(x) + a^\ell \mathbf{1} \right) + b^\ell \mathbf{1} \\ &= R^\ell * f^{\ell-1}(x) + K^{2\ell} * K^{2\ell-1} * f^{\ell-1}(x) - R^\ell * f^{\ell-1}(x) + B^\ell * a^\ell \mathbf{1} + b^\ell \mathbf{1} \\ &= K^{2\ell} * K^{2\ell-1} * f^{\ell-1}(x) + B^\ell * a^\ell \mathbf{1} + b^\ell \mathbf{1} \end{aligned} \tag{4.19}$$

for $\ell = 1 : L-1$.

Next, we show how to define B^L. First, let us denote $f_N(\mathcal{V}(x)) = \alpha \cdot \sigma\left(Wx + \beta \right)$ as the fully connected neural network approximation for $f(x)$ as in Theorem 3. In addition, we denote $\overline{f}(x) = \overline{a} \cdot \overline{f}^{d/2}(x) = f_N(x)$ as the CNN approximation of $f(x)$ as in Theorem 4. Now, we can take

$$[a]_n = \begin{cases} [\overline{a}]_n, & \text{if } n \le Nd^2, \\ -1, & \text{if } n = Nd^2 + (d/2)^2 \text{ or } (N+1)d^2 + (d/2)^2, \\ 0, & \text{others}. \end{cases} \tag{4.20}$$

Recalling that $f^{L-1}(x)$ in the pre-act ResNet now is a linear function, we have

$$a \cdot \mathcal{V}\left(\left[R^L * f^{L-1}(x)\right]\right) = h \cdot \mathcal{V}(x) + c, \tag{4.21}$$

where $h \in \mathbb{R}^{d^2}$ and $c \in \mathbb{R}$. Then, we can redefine a new fully connected neural network function

$$f_{N+2} := \alpha \cdot \sigma\left(W\mathcal{V}(x) + \beta\right) + \sigma(h \cdot \mathcal{V}(x) + c) + \sigma(-h \cdot \mathcal{V}(x) - c). \tag{4.22}$$

According to Lemma 4, we have a CNN function $\widehat{f}(x)$ such that

$$[\widehat{f}^{d/2}(x)]_{:,d/2,d/2} = \left[\sigma(\widehat{K}^{d/2} * \widehat{f}^{d/2-1} + \widehat{b}^{\ell})\right]_{:,d/2,d/2} = \begin{pmatrix} \sigma\left(W\mathcal{V}(x) + \beta\right) \\ \sigma(h \cdot \mathcal{V}(x) + c) \\ \sigma(-h \cdot \mathcal{V}(x) - c) \end{pmatrix}. \tag{4.23}$$

As noticed before, K^{ℓ} for $\ell = 1 : d/2 - 1$ have the same structure in Lemma 4. As a result, we have

$$\widehat{f}^{d/2-1}(x) = \overline{f}^{d/2-1}(x), \quad \forall x \in \Omega. \tag{4.24}$$

Then, we take

$$K^L = \widehat{K}^{d/2}, \tag{4.25}$$

which implies that

$$\begin{aligned} f^L(x) &= R^L * f^{L-1}(x) + \sigma\left(B^L * \sigma\left(A^L * f^{L-1}(x) + a^L\mathbf{1}\right) + b^L\mathbf{1}\right) \\ &= R^L * f^{L-1}(x) + \sigma\left(K^{2L} * K^{2L-1} * f^{L-1}(x) + B + b^L\mathbf{1}\right) \\ &= R^L * f^{L-1}(x) + \sigma\left(\widehat{K}^{d/2} * K^{d/2-1} * \cdots * K^1 * x + B + b^L\mathbf{1}\right), \end{aligned} \tag{4.26}$$

where B is a constant tensor similar to the case described in Theorem 5. Thus, we can take

$$[b^L]_n = \begin{cases} [\beta]_n - [B]_{n,d/2,d/2}, & \text{if } n \le N, \\ c - [B]_{N+1,d/2,d/2}, & \text{if } n = N+1, \\ -c - [B]_{N+2,d/2,d/2}, & \text{if } n = N+2, \end{cases} \tag{4.27}$$

which leads to

$$
\begin{aligned}
&\left[f^L(x)\right]_{:,d/2,d/2} \\
&\quad = \left[R^L * f^{L-1}(x) + \sigma\left(\widehat{K}^{d/2} * K^{d/2-1} * \cdots * K^1 * x + B + b^L \mathbf{1}\right)\right]_{:,d/2,d/2} \\
&\quad = \left[R^L * f^{L-1}(x) + \sigma\left(\widehat{K}^{d/2} * K^{d/2-1} * \cdots * K^1 * x + \widehat{\beta}\mathbf{1}\right)\right]_{:,d/2,d/2} \\
&\quad = \left[R^L * f^{L-1}(x)\right]_{:,d/2,d/2} + \begin{pmatrix} \sigma\left(W\mathcal{V}(x) + \beta\right) \\ \sigma(h \cdot \mathcal{V}(x) + c) \\ \sigma(-h \cdot \mathcal{V}(x) - c) \end{pmatrix},
\end{aligned}
\tag{4.28}
$$

where $\widehat{\beta} = (\beta, c, -c)$. Noticing that $\sigma(x) + \sigma(-x) = x$, finally we check that

$$
\begin{aligned}
\widetilde{f}(x) &= a \cdot \mathcal{V}(f^L)(x) \\
&= a \cdot \mathcal{V}\left(R^L * f^{L-1}(x)\right) + (\alpha, -1, -1) \cdot \begin{pmatrix} \sigma\left(W\mathcal{V}(x) + \beta\right) \\ \sigma(h \cdot \mathcal{V}(x) + c) \\ \sigma(-h \cdot \mathcal{V}(x) - c) \end{pmatrix} \\
&= l(x) + f_N(\mathcal{V}(x)) - \left(\sigma\left(l(x)\right) + \sigma\left(-l(x)\right)\right) \\
&= f_N(\mathcal{V}(x)), \quad \forall x \in \Omega,
\end{aligned}
\tag{4.29}
$$

where $l(x) = h \cdot \mathcal{V}(x) + c$. This completes the proof. □

The approximation property of MgNet. First, we introduce a typical MgNet [13,14] network used in image classification.

Algorithm 1 $u^J = \mathrm{MgNet}(f)$

```
Input: number of grids J, number of smoothing iterations $\nu_\ell$ for $\ell = 1 : J$, number of channels $c_{f,\ell}$ for $f^\ell$
and $c_{u,\ell}$ for $u^{\ell,i}$ on $\ell$th grid.
Initialization: $f^1 = f_{\mathrm{in}}(f)$, $u^{1,0} = 0$
for $\ell = 1 : J$ do
    for $i = 1 : \nu_\ell$ do
        Feature extraction (smoothing):
            $u^{\ell,i} = u^{\ell,i-1} + \sigma \circ B^{\ell,i} * \sigma\left(f^\ell - A^\ell * u^{\ell,i-1}\right) \in \mathbb{R}^{c_{u,\ell} \times n_\ell \times m_\ell}$.    (4.30)
    end for
    Note: $u^\ell = u^{\ell,\nu_\ell}$
    Interpolation and restriction:
        $u^{\ell+1,0} = \Pi_\ell^{\ell+1} *_2 u^\ell \in \mathbb{R}^{c_{u,\ell+1} \times n_{\ell+1} \times m_{\ell+1}}$.    (4.31)
        $f^{\ell+1} = R_\ell^{\ell+1} *_2 (f^\ell - A^\ell(u^\ell)) + A^{\ell+1} * u^{\ell+1,0} \in \mathbb{R}^{c_{f,\ell+1} \times n_{\ell+1} \times m_{\ell+1}}$.    (4.32)
end for
```

Here, $\Pi_\ell^{\ell+1} *_2$ and $R_\ell^{\ell+1} *_2$ in (4.31) and (4.32), respectively, which work as the coarsening operation, are defined as convolutions with stride two [8]. As we do not include coarsening (subsampling or pooling) layers in this study, we propose the following version of the feature extraction (smoothing) iteration in MgNet to study its approximation property.

MgNet:

$$\begin{cases} f^{\ell}(x) = R^{\ell} * f^{\ell-1}(x) + \sigma\left(B^{\ell} * \sigma\left(\theta^{\ell} * x - A^{\ell} * f^{\ell-1}(x) + a^{\ell}\mathbf{1}\right) + b^{\ell}\mathbf{1}\right), \\ f(x) = a \cdot \mathcal{V}(f^{L}(x)), \end{cases} \tag{4.33}$$

for $\ell = 1 : L$, where $f^0(x) = x \in \mathbb{R}^{d\times d}$, $A^{\ell} \in \mathbb{R}^{c_{\ell-1}\times C_{\ell}\times 3\times 3}$, $B^{\ell} \in \mathbb{R}^{C_{\ell}\times c_{\ell}\times 3\times 3}$, $R^{\ell} \in \mathbb{R}^{c_{\ell-1}\times c_{\ell}\times 1\times 1}$, $a^{\ell} \in \mathbb{R}^{C_{\ell}}$, $\theta^{\ell} \in \mathbb{R}^{1\times C_{\ell}\times 3\times 3}$, $b^{c_{\ell}}$, and $a \in \mathbb{R}^{c_L d^2}$.

As discussed in relation to ResNet and pre-act ResNet, we add extra 1×1 convolutional kernels R^{ℓ} and θ^{ℓ} in case of a change of channel. Given that we can recover pre-act ResNet by taking $\theta^{\ell} = 0$ in (4.33), we have the following theorem pertaining to the approximation property of MgNet.

Theorem 7 *Let $\Omega \subset \mathbb{R}^{d\times d}$ be bounded, and assume that $f : \Omega \mapsto \mathbb{R}$ with $\|f\|_{\mathscr{K}_1(\mathbb{D})} < \infty$. Consider $\widetilde{f}(x)$ as an MgNet in (4.33) with any $R^{\ell} \in \mathbb{R}^{c_{\ell-1}\times c_{\ell}\times 1\times 1}$ given a prior, and hyperparameters that satisfy*

$$\begin{aligned} &\textit{depth (number of blocks):} && L = \lfloor \lfloor d/2 \rfloor /2 \rfloor, \\ &\textit{width (number of channels):} && c_{\ell} = (4\ell+1)^2,\ C_{\ell} = 2(4\ell-1)^2, \end{aligned} \tag{4.34}$$

for $\ell = 1 : L-1$, $C_L = 2(4L-1)^2$, and $c_L = N+2$, such that

$$\|f - \widetilde{f}\|_{L^2(\Omega)} \lesssim N^{-\frac{1}{2}-\frac{3}{2d^2}} \|f\|_{\mathscr{K}_1(\mathbb{D})}. \tag{4.35}$$

5 Conclusion

By carefully studying the decomposition theorem for convolutional kernels with large spatial size, we obtained the universal approximation property for a typical deep ReLU CNN structure. In general, we proved that deep multi-channel ReLU CNNs can represent one-hidden-layer ReLU NNs. Consequently, this representation result provides the same asymptotic approximation rate for deep ReLU CNNs as for one-hidden-layer ReLU NNs. Moreover, we established approximation results for one version of ResNet, pre-act ResNet, and MgNet, based on the connections between these commonly used CNN models. This study provides new evidence of the theoretical foundation of classical CNNs and popular architecture, such as ResNet, pre-act ResNet, and MgNet.

Although the approximation properties do not show that CNNs should work better than DNNs in terms of approximation power, this study furthers the fields in understanding of CNNs in a significant way. For example, the success of CNNs may imply that the $\|f\|_{\mathscr{K}_1(\mathbb{D})}$ norm is very small for the target feature extraction function f in image classification or that f belongs to a very special function class that can be efficiently represented or approximated by CNNs efficiently. In addition, we anticipate that it will be possible to apply this kind of approximation results in designing new CNN structures, especially in the context of scientific computing [19]. Furthermore, as the pooling operation plays a key role in practical CNNs, a natural future direction proceeding from this study is to derive approximation results for CNNs involving pooling layers.

Acknowledgements

All the authors were partially supported by the Center for Computational Mathematics and Applications (CCMA) at The Pennsylvania State University. The first author was also supported by the R.H. Bing Fellowship from The University of Texas at Austin. In addition, the third author was supported by the Verne M. William Professorship Fund from The Pennsylvania State University and by the National Science Foundation (Grant No. DMS-1819157 and DMS-2111387).

Data availability

No datasets were generated or analyzed during the current study.

Author details

[1]Department of Mathematics, The University of Texas at Austin, Austin, TX 78712, USA, [2]Beijing International Center for Mathematical Research, Peking University, Beijing 100871, China, [3]Department of Mathematics, The Pennsylvania State University, University Park, PA 16802, USA.

Received: 31 August 2021 Accepted: 25 May 2022
Published online: 30 June 2022

References

1. Arora, R., Basu, A., Mianjy, P., Mukherjee, A.: Understanding deep neural networks with rectified linear units. In: International Conference on Learning Representations (2018)
2. Bach, F.: Breaking the curse of dimensionality with convex neural networks. J. Mach. Learn. Res. **18**(1), 629–681 (2017)
3. Bao, C., Li, Q., Shen, Z., Tai, C., Wu, L., Xiang, X.: Approximation analysis of convolutional neural networks. Work **65**, 871 (2014)
4. Barron, A.R.: Universal approximation bounds for superpositions of a sigmoidal function. IEEE Trans. Inf. Theor. **39**(3), 930–945 (1993)
5. Cybenko, G.: Approximation by superpositions of a sigmoidal function. Math. Control, Signals Syst. **2**(4), 303–314 (1989)
6. Daubechies, I.: Ten lectures on wavelets. SIAM (1992)
7. Ma, C., Wu, L.: The barron space and the flow-induced function spaces for neural network models. Construct. Approx. **87**, 1–38 (2021)
8. Goodfellow, I., Bengio, Y., Courville, A.: Deep learning. MIT Press, Cambridge (2016)
9. Gühring, I., Kutyniok, G., Petersen, P.: Error bounds for approximations with deep relu neural networks in $w^{s,p}$ norms. Anal. Appl. **18**(05), 803–859 (2020)
10. Guo, X., Li, W., Iorio, F.: Convolutional neural networks for steady flow approximation. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp. 481–490 (2016)
11. He, J., Li, L., Xu, J.: Relu deep neural networks from the hierarchical basis perspective. arXiv preprint http://arxiv.org/abs/2105.04156 (2021)
12. He, J., Li, L., Xu, J., Zheng, C.: Relu deep neural networks and linear finite elements. J. Comput. Math. **38**(3), 502–527 (2020)
13. He, J., Xu, J.: Mgnet: a unified framework of multigrid and convolutional neural network. Sci. China Math. **65**, 1–24 (2019)
14. He, J., Xu, J., Zhang, L., Zhu, J.: An interpretive constrained linear model for resnet and mgnet. http://arxiv.org/abs/2112.07441 (2021)
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778 (2016)
16. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: European conference on computer vision, pp. 630–645. Springer (2016)
17. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Netw. **2**(5), 359–366 (1989)
18. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4700–4708 (2017)
19. Karniadakis, G.E., Kevrekidis, I.G., Lu, L., Perdikaris, P., Wang, S., Yang, L.: Physics-informed machine learning. Nature Rev. Phys. **3**(6), 422–440 (2021)
20. Klusowski, J.M., Barron, A.R.: Approximation by combinations of relu and squared relu ridge functions with ℓ^1 and ℓ^0 controls. IEEE Trans. Inf. Theory **64**(12), 7649–7656 (2018)
21. Kohler, M., Langer, S.: Statistical theory for image classification using deep convolutional neural networks with cross-entropy loss. http://arxiv.org/abs/2011.13602 (2020)
22. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Adv. Neural Inf. Process. Syst. **25**, 1097–1105 (2012)
23. Kumagai, W., Sannai, A.: Universal approximation theorem for equivariant maps by group cnns. http://arxiv.org/abs/2012.13882 (2020)
24. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015)
25. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
26. Leshno, M., Lin, V.Y., Pinkus, A., Schocken, S.: Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. Neural Netw. **6**(6), 861–867 (1993)
27. Lin, S.B., Wang, K., Wang, Y., Zhou, D.X.: Universal consistency of deep convolutional neural networks. http://arxiv.org/abs/2106.12498 (2021)
28. Lu, Z., Pu, H., Wang, F., Hu, Z., Wang, L.: The expressive power of neural networks: a view from the width. Adv. Neural Inf. Process. Syst. **36**, 6231–6239 (2017)

29. Montufar, G.F., Pascanu, R., Cho, K., Bengio, Y.: On the number of linear regions of deep neural networks. Adv. Neural Inf. Process. Syst. **587**, 2924–2932 (2014)
30. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on International Conference on Machine Learning, pp. 807–814 (2010)
31. Oono, K., Suzuki, T.: Approximation and non-parametric estimation of resnet-type convolutional neural networks. In: International Conference on Machine Learning, pp. 4922–4931. PMLR (2019)
32. Opschoor, J.A., Petersen, P.C., Schwab, C.: Deep relu networks and high-order finite element methods. Anal. Appl. **45**, 1–56 (2020)
33. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: an imperative style, high-performance deep learning library. Adv. Neural Inf. Process. Syst. **32**, 8026–8037 (2019)
34. Petersen, P., Voigtlaender, F.: Equivalence of approximation by convolutional neural networks and fully-connected networks. Proc. Am. Math. Soc. **148**(4), 1567–1581 (2020)
35. Poggio, T., Mhaskar, H., Rosasco, L., Miranda, B., Liao, Q.: Why and when can deep-but not shallow-networks avoid the curse of dimensionality: a review. Int. J. Autom. Comput. **14**(5), 503–519 (2017)
36. Shen, Z., Yang, H., Zhang, S.: Nonlinear approximation via compositions. Neural Netw. **119**, 74–84 (2019)
37. Siegel, J.W., Xu, J.: Approximation rates for neural networks with general activation functions. Neural Netw. **128**, 313–321 (2020)
38. Siegel, J.W., Xu, J.: Characterization of the variation spaces corresponding to shallow neural networks. http://arxiv.org/abs/2106.15002 (2021)
39. Siegel, J.W., Xu, J.: Improved approximation properties of dictionaries and applications to neural networks. http://arxiv.org/abs/2101.12365 (2021)
40. Siegel, J.W., Xu, J.: High-order approximation rates for shallow neural networks with cosine and reluk activation functions. Appl. Comput. Harmonic Anal. **58**, 1–26 (2022)
41. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International Conference on Machine Learning, pp. 6105–6114. PMLR (2019)
42. Telgarsky, M.: Benefits of depth in neural networks. J. Mach. Learn. Res. **49**, 1517–1539 (2016)
43. Xu, J.: Finite neuron method and convergence analysis. Commun. Comput. Phys. **28**(5), 1707–1745 (2020)
44. Yarotsky, D.: Error bounds for approximations with deep relu networks. Neural Netw. **94**, 103–114 (2017)
45. Zhou, D.X.: Deep distributed convolutional neural networks: Universality. Anal. Appl. **16**, 895–919 (2018)
46. Zhou, D.X.: Universality of deep convolutional neural networks. Appl. Comput. Harmonic Anal. **48**(2), 787–794 (2020)

Publisher's Note