Maximum-and-Concatenation Networks

Xingyu Xie [1] **Hao Kong** [1] **Jianlong Wu** [2] **Wayne Zhang** [3] **Guangcan Liu** [4] **Zhouchen Lin** [1]

Abstract

While successful in many fields, deep neural networks (DNNs) still suffer from some open problems such as bad local minima and unsatisfactory generalization performance. In this work, we propose a novel architecture called Maximum-and-Concatenation Networks (MCN) to try eliminating bad local minima and improving generalization ability as well. Remarkably, we prove that MCN has a very nice property; that is, *every local minimum of an $(l+1)$-layer MCN can be better than, at least as good as, the global minima of the network consisting of its first l layers*. In other words, by increasing the network depth, MCN can autonomously improve its local minima's goodness, what is more, *it is easy to plug MCN into an existing deep model to make it also have this property*. Finally, under mild conditions, we show that MCN can approximate certain continuous functions arbitrarily well with *high efficiency*; that is, the covering number of MCN is much smaller than most existing DNNs such as deep ReLU. Based on this, we further provide a tight generalization bound to guarantee the inference ability of MCN when dealing with testing samples.

1. Introduction

Deep neural networks (DNNs) have been showing superior performance in various fields such as computer vision, speech recognition, natural language processing, and so on. At the first glance, DNN learning is not an enigmatic technique, as its basic idea is quite simple and mostly about learning a possibly over-parameterized DNN from a huge number of training samples; namely,

$$\min_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) := \frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_i), \mathbf{y}_i), \tag{1}$$

where $\mathbf{x}_i \in \mathbb{R}^{d_x}$ and $\mathbf{y}_i \in \mathbb{R}^{d_y}$ denote an input and a target, respectively, $\mathbf{f}_{\boldsymbol{\theta}}(\cdot)$ standards for a DNN with parameters $\boldsymbol{\theta}$, and $\ell(\cdot)$ is some loss function. Notice that, some kind of regularization schema has already been implanted into the network to constrain the parameter space, though there is no explicit regularizer imposed on $\boldsymbol{\theta}$ (Arora et al., 2019a). Despite its ordinary appearance, DNN learning is meanwhile quite complicated in many ways, and the current DNNs still suffer from several weaknesses, e.g., the training procedure may easily get stuck in *bad local minima* (i.e., the local minima with large training error), the learnt model may be prone to *over-fit the training data* (i.e., the testing error is large when small training error is obtained), etc. Overcoming these difficulties are crucial for DNNs to solve the real-world problems that are more challenging and significant, but they are still *open* problems.

To address the issue of bad local minima, many heuristic techniques have been proposed, e.g., batch normalization (Ioffe & Szegedy, 2015), group normalization (Wu & He, 2018), dropout (Srivastava et al., 2014), etc. These techniques would be useful under certain context, but may not be generally helpful and, even worse, it is hard to know when and which method should be used. In fact, the elimination of bad local minima, i.e., having small empirical training error at *all* local minima, is really important for DNN learning. Some recent theories (Zhang et al., 2017; Wei & Ma, 2019; Cao & Gu, 2019; Li & Liang, 2018; Allen-Zhu et al., 2018; Arora et al., 2019c) have revealed that, whenever the local minima produces only small training error, DNNs have probably good generalization performance at these local minima. That is to say, in some cases, good local minima mean good predictors which are the ultimate goal of supervised learning. With the hope of pursuing the property of *no bad local minima*, some learning theories (Kawaguchi, 2016; Arora et al., 2018; Hardt & Ma, 2016; Liang et al., 2018a;b) have been established to prove that, under certain conditions, any local minima of a certain DNN are also global minima. While impressive, existing studies are still unsatisfactory in some aspects:

- Most existing theories about "all local minima are

[1]Key Lab. of Machine Perception (MoE), School of EECS, Peking University [2]School of Computer Science and Technology, Shandong University [3]SenseTime Research [4]B-DAT and CICAEET, School of Automation, Nanjing University of Information Science and Technology. Correspondence to: Guangcan Liu <gcliu@nuist.edu.cn>, Zhouchen Lin <zlin@pku.edu.cn>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

global minima" are built upon on some unrealistic network architectures, e.g., without activation function, which means that they cannot be applied to common deep learning tasks. The work (Kawaguchi & Kaelbling, 2019) considers general architectures, but requires additional regularizer and is limited to shallow case. In addition, strictly speaking, the conclusion of "all local minima are global minima" cannot really ensure that "DNN has no bad local minima". This is because, whenever the adopted network itself is poorly designed, global minima can still lead to large training error. In one word, existing studies have not gained convenient schemes that can be easily used to reduce the training error of general DNNs.

- Though small training error may bring good generalization for some specially designed DNNs (Zhang et al., 2017; Wei & Ma, 2019; Cao & Gu, 2019; Li & Liang, 2018; Allen-Zhu et al., 2018; Arora et al., 2019c), a rigorous generalization bound is still important for general DNNs to produce superior performance in practice. There is sparse research in the direction of generalization analysis, e.g., deep ReLU (Yarotsky, 2017). However, the covering number in deep ReLU is very large, which means that the approximation ability of the network is rather weak.

- What is more, to our knowledge, there is no theoretical study that addresses the issues of local minima and generalization ability simultaneously. These two problems are closely related and should be investigated at the same time.

To relieve the issues highlighted above, we propose a novel multi-layer DNN termed **M**aximum-and-**C**oncatenation **N**etworks (MCN). In our MCN, one hidden layer is formed by concatenating together two parts, with one being a linear transformation of the output of the previous layer, and the other being a maximum of two piecewise smooth functions. The output of the final layer is further transformed by some linear operators, so as to stay in step with the configuration of the target output. In general, the concatenation operator is a good option during designing DNNs, and it is indeed a primary cause of the superiorities of MCN over existing architectures.

We prove that MCN naturally ensures the effectiveness of its learning process, i.e., the no bad local minima property. To be more precise, suppose that $\boldsymbol{\theta}'$ is a global minimum to (1) with $\mathbf{f}_{\boldsymbol{\theta}'}$ being an l-layer MCN (briefly, we say that $\boldsymbol{\theta}'$ is a global minimum of an l-layer MCN), and $\boldsymbol{\theta}$ is a local minimum of the $(l+1)$-layer MCN obtained by adding one layer to the former l-layer network. Then we have $L(\boldsymbol{\theta}) \leq L(\boldsymbol{\theta}')$, which means that *the global minima of an l-layer MCN may be outperformed, at least can be attained, by simply increasing the network depth*. More importantly, MCN can be easily appended to many existing network architectures, and we prove that, under mild conditions, *the modified DNN will get the nice properties of MCN*. This property is achieved mainly due to a *skip connection* with a proper activation function: With the help of this skip connection, the bad local minima are moved to infinity, while the implicit regularizer carried by the network itself may encourage the optimization procedure to seek for the remaining good local minima.

Notice that, piecewise liner functions can approximate any Lipschitz continuous function up to arbitrarily small error, and the maximum operator can model the piecewise linear function efficiently (Telgarsky, 2016). Based on these facts, we show that MCN with *sparse* connection can approximate a wide range of continuous functions arbitrarily well. Our analysis framework is new and quite different from the previous studies (Lu et al., 2020; Yarotsky, 2018; 2017), which rely on Taylor expansion and requires a parameter complexity of $\mathcal{O}(N^{d_x})$, where $N \gg 1$ is a quantity that controls the approximation accuracy [1]. By sharp contrast, we show that a complexity of only $\mathcal{O}(N(\ln N)^{d_x-2})$ is enough to approximate the target function.

Based on the approximation analysis, we further investigate the generalization ability of MCN to cope with testing samples, proving that MCN has much smaller covering number than deep ReLU. Interestingly, our results suggest that the width has less effects than the depth on the generalization bound. Our results also show that, whenever the training data are exactly fitted, *MCN achieves the statistically optimal rate in the minmiax sense*; this confirms the conjectures in (Wei & Ma, 2019; Arora et al., 2019c; Belkin et al., 2018b) that ultra-deep networks may generalize well on testing data [2]. To summarize, the contributions of this paper mainly include:

- We propose a novel architecture termed MCN and prove that MCN can help to overcome the issue of bad local minima. Namely, the global minima of an l-layer MCN can be always attained or even outperformed by simply increasing the network depth (Theorem 1). More importantly, we show that MCN is able to turn a possibly poorly-designed DNN into a good one, which also has the nice property of "no bad local minima" under certain conditions (Corollary 4.2 and Theorem 5). These results would be more significant than (Kawaguchi, 2016; Hardt & Ma, 2016), which only show that all local minima of a certain DNN with fixed depth are global minima, but provide no practical guidance for the users to seek better solutions to their

[1] $N^{-\beta}$ is the dominant term in approximation error, where $\beta > 0$ relates to the smoothness of the target function.

[2] Note here that we have no intention to suggest using infinitely deep networks, as the computational cost is also a matter and the required data amount in the extreme case could be huge.

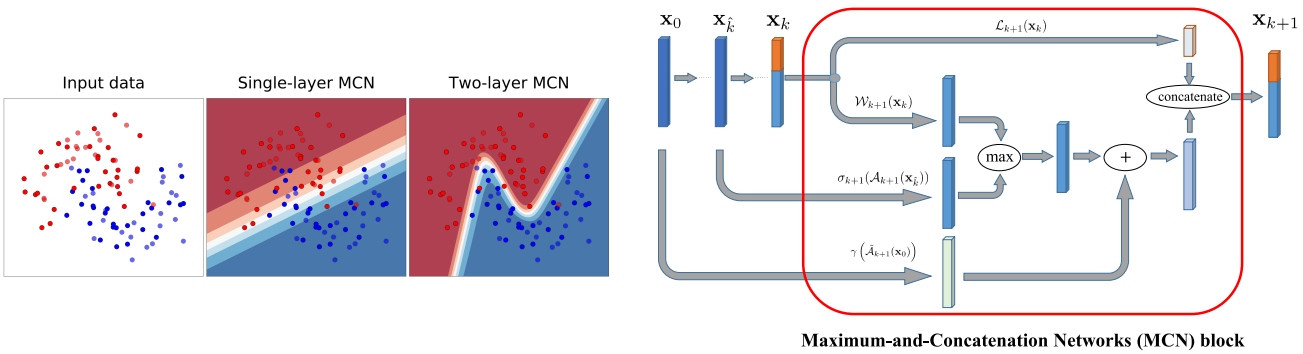

Figure 1. Left: Illustration of the motivations for inventing MCN, which is indeed a generalization of the piecewise smooth function. The composition of MCNs may increase the pieces exponentially. Right: One block of MCN, where each layer consists of four parts.

tasks—just finding the globally optimal solutions to some over-simplified optimization problems is essentially not enough.

- We devise a new framework to analyze the approximation ability of MCN, showing that MCN can approximate some classes of continuous functions arbitrarily well by only using a parameter complexity of $\mathcal{O}(N(\ln N)^{d_x-2})$ (Theorem 2). This is much lower than the $\mathcal{O}(N^{d_x})$ complexity obtained by the previous studies (Lu et al., 2020; Yarotsky, 2018; 2017).
- Unlike the previous analyses in (Liang et al., 2018a;b; Kawaguchi & Kaelbling, 2019), which focus on the elimination of local minima but ignore the generalization performance, we provide rigorous analysis to guarantee the generalization ability of MCN under certain conditions (Theorem 3 and Corollary 3.1). In particular, our results show that MCN has a much smaller covering number than deep ReLU, revealing that the depth is more important than the width for generalization; this supports the mechanism of deep learning.

2. Model and Setting

This section introduces the technical details of MCN, as well as the setup for establishing theoretical analysis.

2.1. Maximum-and-Concatenation Networks

The design of our MCN—a linearity and maximum concatenation network—is inspired by the following observations. Consider the task of shattering some points that are not linearly separable, which is shown in Figure 1. Intuitively, the maximum of two hyperplanes may produce smaller classification error than every single one of them. Therefore, we may reduce the classification error by replacing parts of the current classifier with some maximum-derived units. Such a replacement process can be repeated several times, learning progressively a refined classification surface that will be piecewise smooth. Moreover, considering the regression problem, we have a classical claim from the Stone-Weierstrass approximation theorem.

Claim 1. *Any Lipschitz continuous function can be approximated arbitrarily well by a piecewise linear function.*

By composing a series of maximum operators, we can easily construct a piecewise smooth function. Consider approximating the quadratic function $x \to x^2$. Define the operator $\mathcal{T}^m(x) := \max\{-x/2, x/2 - 2^{1-2m}\}$ and let $g^m(x) := \mathcal{T}^m \circ \mathcal{T}^{m-1} \circ \cdots \mathcal{T}^1(x)$. It is known that $x + \sum_{i=1}^{m} g^i(x)$ approximates x^2 exponentially fast in m (Telgarsky, 2016). In contrast, to approximate a twice differentiable non-piecewise linear function f, it would be awkward to use some existing DNNs that need to rescale the second order differences: $(f(t+2\delta x) - 2f(t+x\delta) + f(x\delta))/(\delta^2 f''(t)) \to x^2$ for $\delta \to 0$ with $f''(t) \neq 0$. Note that $\delta \to 0$ will cause the scale of network parameters to be very large.

Beneath it all, the model of an l-layer MCN, which is indeed a mapping from input $\mathbf{x}$ to output $\mathbf{y}$, is designed as follows, for $k = 0, \cdots, l-1$:

$$\mathbf{x}_{k+1} = \left[\mathcal{L}_{k+1}(\mathbf{x}_k);\ \gamma\left(\tilde{\mathcal{A}}_{k+1}(\mathbf{x}_0)\right) + \mathcal{M}_{k+1}(\mathbf{x}_k)\right], \tag{2}$$

where

$$\mathcal{M}_{k+1}(\mathbf{x}_k) = \max\left\{\mathcal{W}_{k+1}(\mathbf{x}_k), \sigma_{k+1}\left(\mathcal{A}_{k+1}\left(\mathbf{x}_{\hat{k}}\right)\right)\right\},$$

$0 \leq \hat{k} \leq k$ ($\mathbf{x}_{\hat{k}}$ is the output of any intermediate layer between $\mathbf{x}_k$ and $\mathbf{x}_0$), $\gamma(\cdot)$ and $\sigma_{k+1}(\cdot)$ are some element-wise activation functions, $\mathbf{x}_0 = \mathbf{x} \in \mathbb{R}^{d_x}$ is the input data vector, $\mathbf{x}_k \in \mathbb{R}^{d_k}$ is the output of the k-th layer, $[\,;\,]$ is the operator that vertically concatenates two vectors into a single one, $\mathcal{L}_{k+1} : \mathbb{R}^{d_k} \to \mathbb{R}^{d_{\mathcal{L}}}$ is a learnable linear operator[3],

[3]For convenience, we assume that the output of $\mathcal{L}_{k+1}$ has a

and $\mathcal{A}_{k+1}(\cdot)$, $\tilde{\mathcal{A}}_{k+1}(\cdot)$ and $\mathcal{W}_{k+1}(\cdot)$ are all learnable linear operators from $\mathbb{R}^{d_k}$ to $\mathbb{R}^{d_{k+1}-d_{\mathcal{L}}}$.

In fact, as mentioned in Figure 1, MCN is a generalization of piecewise smooth functions, and it can contain many existing DNNs as special cases, e.g., ResNet, Maxout Network (Goodfellow et al., 2013) and Input Convex Neural Networks (ICNN) (Amos et al., 2017). In MCN, there are layers that directly connect the input $\mathbf{x}_0$ to the hidden units in deeper layers. Such connections are unnecessary for traditional networks, but very important for achieving the nice property of "no bad local minimum" which we will introduce later. The highway with the operator $\mathcal{L}_k$ connects the training loss with the geometric projection residual in the proper setting (Section B in supplementary material), which helps MCN perform well when it goes deeper and wider.

2.2. Setting

To analyze MCN theoretically, we consider a typical task of regression (or classification). Denote by $\mathbf{x} \in \mathbb{R}^{d_x}$ and $\mathbf{y} \in \mathbb{R}^{d_y}$ an input vector and a target, respectively. Let $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$ be a training set consisting of n samples, with $\{\mathbf{x}_i\}_{i=1}^n$ being *distinct* points in $\mathbb{R}^{d_x}$. Denote by $\mathbf{x}_{k,i}$ the output of the k-th layer on the i-th training sample $\mathbf{x}_i$. Notice that MCN is primarily designed to learn some extrinsic structures from the data $\mathbf{x}$, and its outputs may be inconsistent with the target $\mathbf{y}$, e.g., they might have different dimensions. Hence, an additional mapping $\Psi : \mathbb{R}^{d_l} \to \mathbb{R}^{d_y}$ is used to further transform the network outputs, resulting in the following objective function for training an l-layer MCN:

$$L(\boldsymbol{\theta}_l) := \frac{1}{n}\sum_{i=1}^{n} \ell(\Psi(\mathbf{x}_{l,i}), \mathbf{y}_i), \tag{3}$$

where $\ell : \mathbb{R}^{d_y} \times \mathbb{R}^{d_y} \to \mathbb{R}$ is an arbitrary lower-bounded loss function (without losing generality, we assume the lower bound is 0), and $\boldsymbol{\theta}_l = \{\boldsymbol{\theta}(\mathcal{L}_k, \mathcal{W}_k, \tilde{\mathcal{A}}_k, \mathcal{A}_k)\}_{k=1}^l$ is a collection of all learnable parameters with $\boldsymbol{\theta}(\mathcal{L}_k, \mathcal{W}_k, \tilde{\mathcal{A}}_k, \mathcal{A}_k)$ being the parameters of the operators $\mathcal{L}_k, \mathcal{W}_k, \tilde{\mathcal{A}}_k$ and $\mathcal{A}_k$ defined in (2). In our setup, the extra mapping $\Psi(\cdot)$ could be either learnt or fixed [4], while the activation functions $\sigma_k(\cdot)$ and $\gamma(\cdot)$ are always fixed.

To obtain rigorous conclusions, some technical conditions are required. But for the ease of presentation, we would like to present them along with the established theorems.

fixed dimension $d_{\mathcal{L}}$, $\forall k = 0, \cdots, l-1$. Actually, our methods and theories do not need this assumption.

[4] There is no much difference between these two variants, as fixing the last layer of a DNN may cause very little influence (Hoffer et al., 2018).

3. Main Results

This section presents the main results of this paper, including a couple of theories regarding the optimality, fitting ability and generalization performance. All the detailed proofs of these theorems are provided in the supplementary material.

3.1. Effects of Depth

First note that an $(l+1)$-layer MCN is obtained by adding one layer into the network consisting of its first l layers, i.e., $\boldsymbol{\theta}_{l+1} = \{\boldsymbol{\theta}_l, \boldsymbol{\theta}(\mathcal{L}_{l+1}, \mathcal{W}_{l+1}, \tilde{\mathcal{A}}_{l+1}, \mathcal{A}_{l+1})\}$. Under some mild technical conditions, we prove that the training objective (3) is non-increasing, or even monotonically decreasing, as the network goes deeper[5].

Theorem 1 (Effects of Depth)**.** *Let the activation function $\gamma(\cdot)$ be the element-wise $\exp(\cdot)$. Suppose that the loss function $\ell(\cdot)$ in (3) is differentiable and convex. Denote by $\boldsymbol{\theta}_{l+1}$ any local minimum of an $(l+1)$-layer MCN. If $d_{l+1} = d_l$, then the following holds for any fixed injection $\Psi(\cdot)$:*

$$L(\boldsymbol{\theta}_{l+1}) \leq \min_{\boldsymbol{\theta}_l'} L(\boldsymbol{\theta}_l'),$$

where $\boldsymbol{\theta}_l'$ is a global minimum of the l-layer MCN. Moreover, if $\ell(\cdot)$ is strongly convex and there exists i such that $\mathbf{x}_{l+1,i} \neq \mathbf{x}_{l,i}'$, then the inequality is strict, namely $L(\boldsymbol{\theta}_{l+1}) < \min_{\boldsymbol{\theta}_l'} L(\boldsymbol{\theta}_l')$.

The setting of fixing $\Psi(\cdot)$ is to ensure that an $(l+1)$-layer MCN and its l-layer part are comparable. According to the above theorem, the global minima of an l-layer MCN can be attained, or even outperformed, by simply increasing the network depth by one. So, given the context of MCN, increasing network depth can not only "eliminate" local minima, but also help seek good solutions that possess smaller training error, providing a theoretically interpretation for a well-known empirical observation—deeper networks usually lead to better training results.

Among the other things, provided that the loss function is differentiable and strongly convex, we can further prove that the training error is able to go to zero. But the proof needs a key theorem established in the next subsection.

Remark 1: One may worry that there exist decreasing paths to infinity, and the weight may need to diverge to improve the performance of local minima (Sohl-Dickstein & Kawaguchi, 2019). The previous work (Kawaguchi, 2016; Liang et al., 2018a;b) may suffer from this problem, mainly due to their explicit regularization, whose coefficient should decay to zero to ensure the consistency of optimization.

[5] This is not in conflict with the learning-based optimization theories (Xie et al., 2019; Liu et al., 2019), which show that their networks can converge fast and need only a smaller number of layers to solve optimization problems. In fact, empirically, MCN will converge when the network is deep enough.

Hence, it leads to the divergence of some parameters to ensure the scale of output. However, our results hold without requiring any parameter to approach zero or infinity. Furthermore, for the classification problem, this divergence problem can be solved by proper parameter regularization (Liang et al., 2019). But, for the general regression problem, regularization may not work. Fortunately, under the over-parameterized setting, algorithmic analysis (Allen-Zhu et al., 2019; Du et al., 2019a) can entirely avoid the divergence risk. We leave the algorithmic analysis of MCN as our future work.

3.2. Approximation Ability

In general, it is unlikely that all mathematical functions can be approximated by DNNs. The following defines a class of functions which can be well approximately by MCN.

Condition 1. *For $\beta \in \mathbb{N}$, we define a modified β-th Sobolev space on the hypercube $[-1,1]^{d_x}$*

$$\mathcal{H}^{\beta} \coloneqq \left\{ \mathbf{f} : \mathrm{D}^{\boldsymbol{\alpha}} \mathbf{f} \in \mathrm{L}^2\left([-1,1]^{d_x}\right), \forall \boldsymbol{\alpha} : |\boldsymbol{\alpha}|_{\infty} \leq \beta \right\},$$

*where $\boldsymbol{\alpha} = (\alpha_1, \cdots, \alpha_{d_x}) \in \mathbb{N}^{d_x}$ is a multi-index, $\mathrm{D}^{\boldsymbol{\alpha}}$ corresponds to the **weak** derivatives operator $\partial_{x_1}^{\alpha_1} \ldots \partial_{x_{d_x}}^{\alpha_d}$ of order $|\boldsymbol{\alpha}| = \alpha_1 + \cdots + \alpha_{d_x}$ and $|\boldsymbol{\alpha}|_{\infty} = \max\{\alpha_i\}$. It is assumed that the function $\mathbf{f} \in \mathcal{H}^{2\beta+2}$ obeys the homogeneous Neumann boundary conditions up to order β:*

$$\partial_{x_j}^{2r+1} \mathbf{f}\Big|_{\partial\Omega_j} = 0, \quad j = 1, \ldots, d_x, \quad r = 0, \ldots, \beta - 1,$$

where $\partial\Omega_j = \left\{ \mathbf{x} \in [-1,1]^d : x_j = \pm 1 \right\}$ is the boundary.

The above condition depicts a class of continuous functions $\mathbf{f} \in \mathrm{L}^2([-1,1]^{d_x})$ such that $\mathbf{f}$ and its weak derivatives up to a certain order have finite L_2 norm. Note that the Neumann boundary condition of $[-1,1]^{d_x}$ is not harsh, and we can always extend the target function by firstly using the Sine or Cosine functions to introduce the homogeneous Neumann property and then scaling it to the interval $[-1,1]^{d_x}$.

As pointed out by (Barron, 1993), a standard fully connected neural network with enough, possibly infinite, hidden units can approximate any continuous function in compact domain. For MCN, we have an explicit approximation bound to connect the width and depth in a finite fashion.

Theorem 2 (Approximation Ability)**.** *Let $\mathbf{f}$ be a vector-valued function that obeys Condition 1, and let $w \geq 0, p \geq 0, N \gg 1$ be given numbers. Define $N_d = N\left(\ln N\right)^{d_x - 2}$, and denote by $\mathbf{f}_{\boldsymbol{\theta}}$ the output of an MCN. Suppose either $\mathbf{f}_{\boldsymbol{\theta}}$ is of width $\mathcal{O}(N_d d_x w p \ln p)$ and depth $\mathcal{O}(l \ln p + N^2)$, or $\mathbf{f}_{\boldsymbol{\theta}}$ has $\mathcal{O}(d_x w p \ln p)$ width and $\mathcal{O}(N_d l \ln p + N^2 N_d)$ depth. Then $\mathbf{f}$ can be approximated by MCN with proper parameters, in a sense that:*

$$\|\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}) - \mathbf{f}(\mathbf{x})\|_{\infty} \leq \epsilon, \quad \forall \mathbf{x} \in [-1,1]^{d_x},$$

where

$$\epsilon = \mathcal{O}\left(d_x 2^{d_x} p^2 2^{-wl} + N^{-2\beta-2} \left(\ln N\right)^{d_x - 1} \right).$$

The number of non-zero parameters in $\boldsymbol{\theta}$ is in the order of $\mathcal{O}\left(N_d \left(d_x w^2 p \ln p + N^2\right)\right)$.

Proof Sketch. We first construct the shallow MCNs that approximate $\sin(n\pi x)$ and $\cos(n\pi x)$ for different $n \in \mathbb{N}$ exponentially fast. Then we can obtain a multivariate function $\phi_{\mathbf{n}} \coloneqq \prod_{i=1}^{d_1} \sin(n_i \pi x) \prod_{k=d_1+1}^{d_x} \cos(n_k \pi x)$ by an MCN of $\mathcal{O}(\ln d_x)$ depth, where $d_1 \leq d_x$ and $\mathbf{n} \in \mathbb{N}^{d_x}$. Since the set $\{\phi_{\mathbf{n}}, \mathbf{n} \in \mathbb{N}^{d_x}\}$ is a Fourier orthogonal basis for $\mathrm{L}^2([-1,1]^{d_x})$, we can prove that $N(\ln N)^{d_x - 2}$ sub-MCNs suffice to approximate the target function, where $N = \prod_i n_i$. More detailed proofs can be founded in the supplementary material. □

Remarkably, Theorem 2 shows that MCN requires only a parameter complexity of $\mathcal{O}\left(d_x N (\ln N)^{d_x - 2}\right)$ to approximate the target function, which is dramatically lower than the $\mathcal{O}(d_x^{d_x} N^{d_x})$ required by deep ReLU (Yarotsky, 2017). This is mainly benefited from our analysis techniques. Unlike the analyses in (Lu et al., 2020; Yarotsky, 2018), which split the input space into small hyper-cubes and use a local network to approximate the Taylor expansion on those hyper-cubes, our analysis is built upon high-dimensional Fourier expansions and can therefore obtain higher decay rate for the approximation residual. Besides, the special network architecture of MCN is another cause of the advantage of lower complexity. Namely, the maximum operator makes the power of the decay term for approximating underlying polynomial be in the order of width×depth. By contrast, the decay power is just proportional to the depth in deep ReLU.

In summary, Theorem 2 illustrates that MCN with highly sparse connectivity between neurons can produce good approximation performance. This forms good basis for establishing tight generalization bound and eliminating bad local minima, as will be shown soon.

3.3. Generalization Bound

Theorem 2 ensures the existence of a good predictor when MCN goes deeper and wider. Now, one natural question is: does the generalization bound also shrink as the network becomes deeper? To analyze the generalization ability of DNNs or any other learning methods, it is indeed necessary to make some assumptions about the data. In this subsection, we set $d_y = 1$ and assume that $\mathbf{x}_i \in [0,1]^{d_x}$ for $i = 1, \cdots, n$. We consider the nonparametric regression task, i.e., there exists a target oracle function f_0 such that

$$y_i = f_0(\mathbf{x}_i) + \varepsilon_i, \quad i = 1, \cdots, n, \tag{4}$$

where the noise terms ε_i's are assumed to be i.i.d. Gassuian and independent of $\mathbf{x}_i$.

Denote the function class of our MCN as

$$\mathcal{F}(\boldsymbol{\theta}, s) \coloneqq \left\{ f_{\boldsymbol{\theta}} : \mathrm{Supp}\left(\boldsymbol{\theta}\right) < s, \|\boldsymbol{\theta}_k\|_F^2 < \infty, \forall k \leq l \right\},$$

where $\|\boldsymbol{\theta}_k\|_F$ is the Frobenius norm of all the parameters at the k-th layer, and the operator $\mathrm{Supp}(\cdot)$ denotes the support of a set, i.e., $\mathrm{Supp}\left(\boldsymbol{\theta}\right)$ is the number of nonzero parameters in MCN. The boundness assumption of $\mathrm{Supp}\left(\boldsymbol{\theta}\right) < s$ is made on the basis of Theorem 2, which shows that MCN with sparse connections can possess strong approximation ability. For convenience, we consider the case where the structure of $\mathcal{F}(\boldsymbol{\theta}, s)$ is deterministic, i.e., the input layer of $\mathcal{A}_k(\cdot)$ is the same for all MCNs in $\mathcal{F}(\boldsymbol{\theta}, s)$. Denote by $\mathcal{N}\left(\delta, \mathcal{F}(\boldsymbol{\theta}, s), \|\cdot\|_1\right)$ the minimal number of ℓ_1-balls with radius δ that covers $\mathcal{F}(\boldsymbol{\theta}, s)$. The logarithm of $\mathcal{N}\left(\delta, \mathcal{F}(\boldsymbol{\theta}, s), \|\cdot\|_1\right)$ is also called the *covering number* for convenience. For an operator $\mathcal{A}$, $\|\mathcal{A}\|_1$ denotes its ℓ_1 norm induced by the vector ℓ_1 norm, namely $\|\mathcal{A}\|_1 = \max_{\mathbf{x}\neq 0} \frac{\|\mathcal{A}(\mathbf{x})\|_1}{\|\mathbf{x}\|_1}$. Then we have the following theorem to bound the covering number (i.e., $\ln \mathcal{N}\left(\delta, \mathcal{F}(\boldsymbol{\theta}, s), \|\cdot\|_1\right)$).

Theorem 3 (Covering Number of MCN)**.** *Assume that the activation function $\sigma_k(\cdot)$ is ρ_k-Lipschitz and $\rho_k \leq \rho$ for $k = 1, \cdots, l$. Then one block of MCN is κ_k-Lipschitz continuous w.r.t. the input layers and*

$$\kappa_k = \left(1 + \max\{\rho_k, 2\}\|\boldsymbol{\theta}_k\|_1\right),$$

where

$$\|\boldsymbol{\theta}_k\|_1 \coloneqq \max\{\|\tilde{\mathcal{A}}_k\|_1, \|\mathcal{A}_k\|_1, \|\mathcal{W}_k + \mathcal{L}_k\|_1\}.$$

Moreover, we have

$$\ln \mathcal{N}\left(\mathcal{F}(\boldsymbol{\theta}, s), \delta, \|\cdot\|_1\right) \leq \mathcal{O}\left(sl \ln\left(\frac{\rho w \prod_{k=1}^{l} \kappa_k}{\delta}\right)\right),$$

where w and l are the width and depth of MCN, respectively.

The above theorem shows that the covering number of MCN is $\mathcal{O}\left(sl^2 \ln\left(w/\delta\right)\right)$, where $s = \Theta\left(d_x N (\ln N)^{d_x - 2}\right)$. By contrast, to achieve the same approximation accuracy, deep ReLU needs a covering number of $\mathcal{O}\left(s' l \ln\left(s' w^2 l/\delta\right)\right)$, with $s' = \Theta(d_x^{d_x} N^{d_x})$. In the situation of high-dimensional data, i.e., d_x is large, it is clear that MCN has much smaller covering number than deep ReLU, which means that the model complexity of MCN is much lower. Due to this, MCN provably owns good generalization performance, as shown in the following.

Corollary 3.1 (Generalization Bound)**.** *Consider the regression problem in (4) and assume $\max_{\mathbf{x}\in[0,1]^{d_x}} f_0(\mathbf{x}) < \infty$. Let f_{M} be any MCN from $\mathcal{F}(\boldsymbol{\theta}, s)$, and define*

$$\ell_n(f) \colon = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - f(\mathbf{x}_i)\right)^2,$$

$$\Delta_n \coloneqq \mathbb{E}_{f_0}\left[\ell_n(f_{\mathrm{M}}) - \inf_{f\in\mathcal{F}} \ell_n(f)\right],$$

where $\mathbb{E}_{f_0}$ is the expectation taken with respect to the samples generated from the regression model (4). Define

$$\mathrm{dis}(f_{\mathrm{M}}, f_0) \coloneqq \mathbb{E}_{f_0}\left[\left(f_{\mathrm{M}}(\mathbf{x}) - f_0(\mathbf{x})\right)^2\right].$$

Then, we have

$$\mathrm{dis}(f_{\mathrm{M}}, f_0) \leq \mathcal{O}\left(\Delta_n + \inf_{f\in\mathcal{F}} \mathrm{dis}(f, f_0) + \frac{sl^2 \ln\left(wn\right)}{n}\right).$$

This corollary is indeed a direct application of the general statics generalization inequality in (Lu et al., 2020; Yarotsky, 2017). As we can see, the generalization bound depends on three parts, intuitively described as $\varepsilon_1 + \varepsilon_2 + \varepsilon_3$, where ε_1 is the gap from the obtained training loss to the global minimal one, ε_2 is the approximation error, and ε_3 is the covering number. Notably, Theorem 1 provides a way to reduce ε_1, and Theorem 3 ensures that small ε_2 unnecessarily results in large ε_3.

For nonparametric regression with square loss, when the target function f_0 is β-smooth, it is well-known that the statistically optimal estimation rate in terms of data size is $n^{-\frac{2\beta}{2\beta+d_x}}$ (Giné & Nickl, 2016), also called as *minimax estimation rate*. Owning the minimax estimation rate means that the estimator performs the best in the worst case. Interestingly, when the training data is fitted exactly, MCN also owns this property.

Theorem 4 (Minimax Estimation Rate)**.** *Suppose that the density $p(\cdot)$ over some compact set $\mathcal{C}$ satisfies*

$$0 < p_{\min} \leq p(\mathbf{x}) \leq p_{\max}, \quad \forall \mathbf{x} \in \mathcal{C}.$$

Assume that the target function f_0 is β-smooth and let $\ell(\cdot)$ in (3) be the square loss. Denote the final output of our model as $f_{\boldsymbol{\theta}}(\mathbf{x}_i)$, where $\boldsymbol{\theta}$ is the learnable parameters of MCN. If $f_{\boldsymbol{\theta}}(\mathbf{x}_i) = y_i$ for $i = 1, \cdots, n$, then for any data sample $\mathbf{x} \in \mathbb{R}^{d_x}$ located in the support set of p, the output of MCN satisfies the following with high probability:

$$\mathbb{E}_{\mathcal{S}^n}\left[\mathbb{E}_{\varepsilon}\left[\|f_{\boldsymbol{\theta}}\left(\mathbf{x}\right) - f_0\left(\mathbf{x}\right)\|^2 \mid \mathcal{S}^n\right]\right] \leq C n^{-\frac{2\beta}{2\beta+d_x}},$$

where $\mathcal{S}^n = \{(\mathbf{x}_i, y_i))\}_{i=1}^n$ and $C > 0$ is a number depends only on the numerical range of the outputs of MCN.

In general, the above theorem confirms the phenomenon that over-parameterized DNNs may not necessarily cause over-fitting (Belkin et al., 2019; 2018b). For Theorem 4 to hold, the training error has to be reduced to zero. This can actually be accomplished by using the techniques in (Gasca & Sauer, 2000) to link together Theorem 1 and Theorem 2, as will be shown in next subsection.

Remark 2: One may worry about the "exact fitting" assumption may not be satisfied since the noise belongs to an unbounded distribution. The derivatives or weights of DNN may diverge to infinity as $n \to \infty$. However, this may not be a problem and exact fitting can easily happen under mild condition. On the one hand, Gaussian distribution has an exponential decay tail. Thus, we can approximately treat it as bounded. On the other hand, some recent results (Arora et al., 2019c; Du et al., 2019a; Allen-Zhu et al., 2018; Liang et al., 2020; E et al., 2019) show that the DNNs, having universal approximation ability, can easily fit the Gaussian noise without any weight diverging. Even more, exact fitting can happen near the initial state of DNN as long as it is wide or deep enough; the depth or width is in the polynomial order of n. For MCN, we already prove its approximation ability in Theorem 2. Following the same road-map, we can conclude that its parameters do not diverge in the exact fitting case.

Remark 3: We remark that the estimator $f_{\boldsymbol{\theta}}$ does not belong to the β-smooth function class (its smoothness depends on the architecture and activation function). In conclusion, even though $f_{\boldsymbol{\theta}}$ is not β-smooth and fits the data exactly, it attains optimal excess loss rates. We refer the readers to (Rakhlin et al., 2017) for further discussion of optimal rates in non-parametric estimation and statistical learning.

Remark 4: For any $\mathbf{x}_i \in \mathcal{S}^n$,

$$\mathbb{E}_{\epsilon_i}\left[\|f_{\boldsymbol{\theta}}(\mathbf{x}_i) - f_0(\mathbf{x}_i)\|^2 \mid \mathcal{S}^n\right] = 1.$$

However,

$$\mathbb{E}_{\mathcal{S}^n}\left[\mathbb{E}_{\varepsilon}\left[\|f_{\boldsymbol{\theta}}(\mathbf{x}_i) - f_0(\mathbf{x}_i)\|^2 \mid \mathcal{S}^n\right]\right] \to 0, \text{ as } n \to \infty,$$

due to the measure of a specific point is 0.

3.4. No Bad Local Minima

As aforementioned, under mild technical conditions, the training error produced by MCN can be arbitrarily small when the network is deep enough.

Corollary 4.1 (Optimal Training Error)**.** *Suppose that the loss function $\ell(\cdot)$ is differentiable and strongly convex. Denote by $\boldsymbol{\theta}_l$ any local minimum of an l-layer MCN. For any $\epsilon > 0$, there exists a $D \in \mathbb{N}$ such that $L(\boldsymbol{\theta}_l) \leq \epsilon$ holds for any $l > D$.*

The "no bad local minima" property of MCN replies on its special network design, and is unnecessarily true for the other DNNs. In the following, we shall introduce two ways to refine an existing DNN that is possibly poorly designed. The first one is straightforward and simply to treat the output of an existing DNN as the input $\mathbf{x}_0$ to MCN, and the parameters of the existing network are not involved in re-training. In this case, it is easy to obtain the following result:

Corollary 4.2 (Partial Training)**.** *For fixed injection $\Psi(\cdot)$ and an existing l_0-layer DNN with output $\mathbf{h}_0$, construct an l-layer MCN with $\gamma(\cdot)$ being element-wisely exponential and input $\mathbf{x}_0 = \mathbf{h}_0$. If $\mathbf{h}_0$ is an injective function w.r.t. the input $\mathbf{x}$ and the loss $\ell(\cdot)$ is differentiable and strongly convex, then for any $\epsilon > 0$, there exists a large enough $D \in \mathbb{N}$, such that $L(\boldsymbol{\theta}_l) \leq \epsilon$ holds for any local minimum $\boldsymbol{\theta}_l$ with $l \geq D$.*

In above corollary, the existing DNN is assumed to be fixed and MCN is simply applied to its output. Actually, it is also feasible to re-train all the parameters, including the parameters of both the existing network and the appended MCN blocks.

Theorem 5 (Full Training)**.** *For fixed injection $\Psi(\cdot)$ and an existing l_0-layer DNN with output $\mathbf{h}_0$, append an l-layer MCN at its end with $\gamma(\cdot)$ being element-wisely exponential, resulting in a new model $\mathbf{h}_l$. Suppose that the loss $\ell(\cdot)$ is differentiable and strongly convex, and there exist parameters that make $\mathbf{h}_0$ be injective. Then, for any $\epsilon > 0$, there exists a large enough $D \in \mathbb{N}$ such that*

$$\frac{1}{n}\sum_{i=1}^{n} \ell(\Phi(\mathbf{h}_l(\mathbf{x}_i)), \mathbf{y}_i) \leq \epsilon$$

holds at any local minimum $\mathbf{h}_l$ with $l \geq D$.

One may have noticed that monotonic decreasing property in Theorem 1 is not enough to guarantee global minimal training loss. In fact, as aforementioned, Theorem 2 also plays an important role in gaining the above results, and we need use the techniques in (Gasca & Sauer, 2000) to link Theorem 1 and Theorem 2 together.

Remarkably, the above results illustrate that MCN is not just an approach for seeking the global optimal solution to certain optimization problems, but instead a powerful tool for helping seek better solutions to the primary task behind the optimization problems.

3.5. Discussions

There is another interpretation for why MCN can eliminate bad local minimum. When adopting the square loss, we find that the loss in (3) at the local minimum equals to a projection residual obtained by projecting the training data onto a subspace. The subspace is expanded by parameters in the concatenation linear part $\mathcal{L}_k(\cdot)$ for $k = 1, \cdots, l$, which means that the subspace is larger when more independent parameters are contained in the linear branch $\mathcal{L}_k(\cdot)$. On the other hand, large space often brings small projection residual. Please see Section B in the supplementary material for more details.

To summarize, this section establishes a collection of theorems to cope with the problems of bad local minima and generalization issue. More precisely, first, Theorem 1 and Corollary 4.1 reveal the "no bad local minima" property of MCN, and Corollary 4.2 and Theorem 5 extend this property to the other DNNs. Second, Theorem 2 shows the

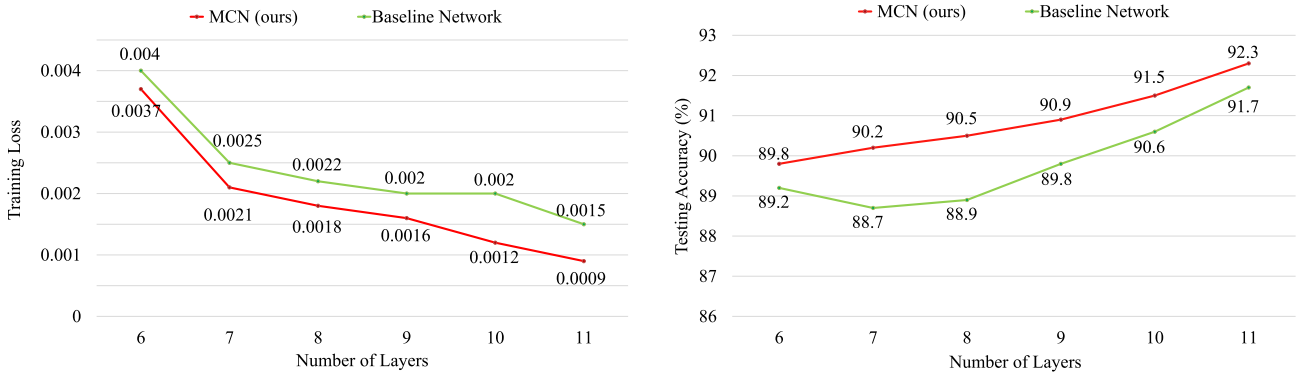

Figure 2. Left: Training loss of our MCN (red) and baseline network (green) with various number of layers. Right: Testing accuracy.

approximation ability of MCN, illustrating that MCN can obtain the same approximation error by using parameters much less than deep ReLU. The number of required parameters is far smaller than the network size, which implies that MCN allows to use some prevalent sparse patterns such as CNN structure and pruning tricks. The sparsity of network connections further leads to a small covering number for MCN in Theorem 3. Based on this, finally, we provide the generalization bound for MCN in Corollary 3.1.

4. Experiments

4.1. Theorems Verification

We conduct experiments on the commonly used CIFAR-10 dataset, with the purpose of validating our theorems as well as the effectiveness of MCN. We first construct a baseline network with 6 weighted layers, including five convolutional layers and one fully-connected layer. Then we add convolutional layers to make the network deeper. It contains five max pooling in total. For our MCN, we replace the convolutional layers after the third max pooling layer with our MCN block. To make a fair comparison, both networks have the same number of layers and parameters, and so for the random seed and learning rate. Also, batch normalization and ReLU are adopted by both networks. For detailed experimental settings and model configurations, please refer to the supplementary material.

Figure 2 shows the training loss and testing accuracy with different number of layers. According to the red line in the left part of Figure 2, the training loss of our MCN monotonically decreases with the increase of depth. This is consistent with our Theorems 1 and 2. From the red line in the right part of Figure 2, we can see that deeper MCN can achieve better testing accuracy, which demonstrates the generalization performance of MCN and confirms our Corollary 3.1 and Theorem 4. In addition, according to the green line in the right part of Figure 2, the testing accuracy of the baseline network does not monotonically increase as the network goes deeper. Therefore, the "no bad local minima" property should be a primary cause of the nice performance of MCN. In summary, compared with the baseline network, our MCN has much lower training loss as well as higher testing accuracy, revealing the superiority of MCN.

4.2. Appending MCN

To validate the merits of Corollary 4.2 and Theorem 5, we add two MCN blocks to VGG19 (Simonyan & Zisserman, 2014) and ResNet18 (He et al., 2016) as the treatment group. The original two architectures, VGG19 and ResNet18, are regarded as the first control group. To make a comparison, we also add two traditional convolutional layers to VGG19 and ResNet18, considered as the second control group. For the treatment group and the second control group, we consider two ways to train the appended VGG19 and ResNet18 (short as Res18). The first one is partial training which treats VGG19 and Res18 as the feature extractors whose parameters are not involved during training. The second one is full training which considers the appended networks as new models and train them from scratch.

Table 1 shows the comparison results among all the three groups, in terms of both training loss and testing accuracy. As we can see, the plugging of traditional convolution layers can decrease the training loss, however, the appending of MCN has more amount of improvement, which, again, show the benefits of the "no bad local minima" property. Interestingly, full training and partial training share comparable performance when appending MCN but not for convolution layers. Hence, both Corollary 4.2 and Theorem 5 are practical theories. Moreover, our MCN outperforms distinctly all the competing methods; this, again, confirms the superiority of our MCN architecture.

Table 1. The training error (Err.) and testing accuracy (Acc.) of different models on the CIFAR-10 dataset. We denote by C the added two convolutional layers and M the appended MCN blocks.

Models	VGG19	VGG19+ C(full)	VGG19+ C(part)	VGG19+ M(full)	VGG19+ M(part)	Res18	Res18+ C(full)	Res18+ C(part)	Res18+ M(full)	Res18+ M(part)
Err.	0.0016	0.0013	0.0015	**0.0010**	0.0011	0.0013	0.0011	0.0012	**0.0009**	**0.0009**
Acc.	92.0%	92.4%	92.1%	**92.8%**	92.6%	92.7%	93.5%	93.1%	93.7%	**93.8%**

Table 2. The training error (Err.) and testing accuracy (Acc.) of different models on the CIFAR-100 dataset. We denote by C the added two convolutional layers and M the appended MCN blocks.

Models	Res18	Res18+ C(full)	Res18+ C(part)	Res18+ M(full)	Res18+ M(part)	ResNeXt29	ResNeXt29+ C(full)	ResNeXt29+ C(part)	ResNeXt29+ M(full)	ResNeXt29+ M(part)
Err.	0.0020	0.0014	0.0015	**0.0009**	**0.0009**	0.0056	0.0051	0.0054	**0.0008**	0.0011
Acc.	76.15%	76.58%	76.48%	**76.95%**	76.87%	80.71%	80.78%	80.69%	**82.31%**	81.41%

4.3. Additional Experiments

To better demonstrate the representation ability of our MCN block, we further conduct some additional experiments on the more complex dataset CIFAR-100, and make comparisons with the SOTA of ResNeXt (Xie et al., 2017) (a more powerful network architecture).

Similar to the previous part, the original two architectures, Res18 and ResNeXt29, are regarded as the first control group. As for the second control group, we still add two traditional convolutional layers to Res18 and ResNeXt29. Besides, we append two MCN blocks to the end of both Res18 and ResNeXt29 as the treatment group.

For the treatment group and the second control group, the two ways to train the appended Res18 and ResNeXt29 remain the same as previous experiment. One is partial training which treats Res18 and ResNeXt29 as the feature extractors, while the other is full training which considers the appended DNNs as new models and train them from scratch.

We present the results of the partial training (i.e., fixing Res18 and ResNeXt29 when appending MCN blocks) in Table 2. It can be seen that, even in the case of handling complex data, our MCN achieves superior results. The treatment groups under two different training methods both outperform the control groups, which is consistent with Table 1. Moreover, by comparing Table 1 with Table 2, our MCN blocks have greatly improved the performance when handling more complex data. Please note that ordinarily appending CNNs cannot ensure the monotonicity of Err. and Acc. This phenomenon not only verifies Corollary 4.2 and Theorem 5 again, but also shows that our MCN has a stronger representation ability than general linear structure, which corresponds to Theorem 2.

5. Conclusion

In this paper, we propose a novel multi-layer DNN structure termed MCN, which can approximate some class of continuous functions arbitrarily well even with highly sparse connection. We prove that the global minima of an l-layer MCN may be outperformed, at least can be attained, by simply increasing the network depth. More importantly, MCN could be easily appended to any of the many existing DNN and the augmented DNN will share the same property of MCN. Finally, we analyze the generalization ability of MCN and reveal that depth is more important than width for generalization; this supports the mechanism of deep learning. In summary, this study does take a step towards the ultimate goal of deep learning theory—to understand why DNNs can work well in a wide variety of applications.

Acknowledgments

This work is supported in part by New Generation AI Major Project of Ministry of Science and Technology of China (grant no 2018AAA0102501), in part by NSF China (grant no.s 61625301 and 61731018), in part by Major Scientific Research Project of Zhejiang Lab (grant no.s 2019KB0AC01 and 2019KB0AB02), in part by Fundamental Research Funds of Shandong University, in part by Future Talents Research Funds of Shandong University, in part by Beijing Academy of Artificial Intelligence, in part by Qualcomm, and in part by SenseTime Research Fund.

References

Adcock, B. Multivariate modified fourier series and application to boundary value problems. *Numerische Mathematik*, 115(4):511–552, 2010.

Allen-Zhu, Z., Li, Y., and Liang, Y. Learning and generalization in overparameterized neural networks, going beyond two layers. *arXiv preprint arXiv:1811.04918*, 2018.

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

Amos, B., Xu, L., and Kolter, J. Z. Input convex neural networks. In *International Conference on Machine Learning*, pp. 146–155, 2017.

Arora, S., Cohen, N., Golowich, N., and Hu, W. A convergence analysis of gradient descent for deep linear neural networks. *arXiv preprint arXiv:1810.02281*, 2018.

Arora, S., Cohen, N., Hu, W., and Luo, Y. Implicit regularization in deep matrix factorization. In *Advances in Neural Information Processing Systems*, pp. 7411–7422, 2019a.

Arora, S., Du, S. S., Hu, W., Li, Z., Salakhutdinov, R., and Wang, R. On exact computation with an infinitely wide neural net. *arXiv preprint arXiv:1904.11955*, 2019b.

Arora, S., Du, S. S., Hu, W., Li, Z., and Wang, R. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *arXiv preprint arXiv:1901.08584*, 2019c.

Barron, A. R. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information theory*, 39(3):930–945, 1993.

Belkin, M., Hsu, D., Ma, S., and Mandal, S. Reconciling modern machine learning and the bias-variance trade-off. *arXiv preprint arXiv:1812.11118*, 2018a.

Belkin, M., Hsu, D. J., and Mitra, P. Overfitting or perfect fitting? risk bounds for classification and regression rules that interpolate. In *Advances in Neural Information Processing Systems*, pp. 2300–2311, 2018b.

Belkin, M., Rakhlin, A., and Tsybakov, A. B. Does data interpolation contradict statistical optimality? In *International Conference on Artificial Intelligence and Statistics*, pp. 1611–1619, 2019.

Cao, Y. and Gu, Q. A generalization theory of gradient descent for learning over-parameterized deep relu networks. *arXiv preprint arXiv:1902.01384*, 2019.

Dou, X. and Liang, T. Training neural networks as learning data-adaptive kernels: Provable representation and approximation benefits. *arXiv preprint arXiv:1901.07114*, 2019.

Du, S. S., Wang, Y., Zhai, X., Balakrishnan, S., Salakhutdinov, R. R., and Singh, A. How many samples are needed to estimate a convolutional neural network? In *Advances in Neural Information Processing Systems*, pp. 373–383, 2018.

Du, S. S., Lee, J. D., Li, H., Wang, L., and Zhai, X. Gradient descent finds global minima of deep neural networks. In *International Conference on Machine Learning*, 2019a.

Du, S. S., Zhai, X., Poczos, B., and Singh, A. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations*, 2019b.

E, W., Ma, C., Wu, L., et al. On the generalization properties of minimum-norm solutions for over-parameterized neural network models. *arXiv preprint arXiv:1912.06987*, 2019.

Gasca, M. and Sauer, T. Polynomial interpolation in several variables. *Advances in Computational Mathematics*, 12 (4):377, 2000.

Giné, E. and Nickl, R. *Mathematical foundations of infinite-dimensional statistical models*, volume 40. Cambridge University Press, 2016.

Goodfellow, I., Warde-Farley, D., Mirza, M., Courville, A., and Bengio, Y. Maxout networks. In *International Conference on Machine Learning*, pp. 1319–1327, 2013.

Györfi, L., Kohler, M., Krzyzak, A., and Walk, H. *A distribution-free theory of nonparametric regression*. Springer Science & Business Media, 2006.

Hardt, M. and Ma, T. Identity matters in deep learning. *arXiv preprint arXiv:1611.04231*, 2016.

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

Hoffer, E., Hubara, I., and Soudry, D. Fix your classifier: the marginal value of training the last weight layer. *arXiv preprint arXiv:1801.04540*, 2018.

Horn, R. A. and Johnson, C. R. Topics in matrix analysis, 1991. *Cambridge University Presss*, 37:39, 1991.

Huybrechs, D., Iserles, A., et al. From high oscillation to rapid approximation iv: Accelerating convergence. *IMA Journal of Numerical Analysis*, 31(2):442–468, 2011.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Kawaguchi, K. Deep learning without poor local minima. In *Advances in Neural Information Processing Systems*, pp. 586–594, 2016.

Kawaguchi, K. and Kaelbling, L. P. Elimination of all bad local minima in deep learning. *arXiv preprint arXiv:1901.00279*, 2019.

Kawaguchi, K., Huang, J., and Kaelbling, L. P. Effect of depth and width on local minima in deep learning. *Neural Computation*, 2019.

Li, Y. and Liang, Y. Learning overparameterized neural networks via stochastic gradient descent on structured data. In *Advances in Neural Information Processing Systems*, pp. 8157–8166, 2018.

Liang, S., Sun, R., Lee, J. D., and Srikant, R. Adding one neuron can eliminate all bad local minima. In *Advances in Neural Information Processing Systems*, pp. 4350–4360, 2018a.

Liang, S., Sun, R., Li, Y., and Srikant, R. Understanding the loss surface of neural networks for binary classification. *arXiv preprint arXiv:1803.00909*, 2018b.

Liang, S., Sun, R., and Srikant, R. Revisiting landscape analysis in deep neural networks: Eliminating decreasing paths to infinity. *arXiv preprint arXiv:1912.13472*, 2019.

Liang, T., Rakhlin, A., and Zhai, X. On the multiple descent of minimum-norm interpolants and restricted lower isometry of kernels. *arXiv preprint arXiv:1908.10292 [cs, math, stat]*, 2020.

Liu, J., Chen, X., Wang, Z., and Yin, W. ALISTA: Analytic weights are as good as learned weights in LISTA. In *International Conference on Learning Representations*, 2019.

Lu, J., Shen, Z., Yang, H., and Zhang, S. Deep network approximation for smooth functions. *arXiv preprint arXiv:2001.03040*, 2020.

Luxburg, U. v. and Bousquet, O. Distance-based classification with lipschitz functions. *Journal of Machine Learning Research*, 5(Jun):669–695, 2004.

Ma, C., Wu, L., et al. A priori estimates of the generalization error for two-layer neural networks. *arXiv preprint arXiv:1810.06397*, 2018.

Maillard, O. and Munos, R. Compressed least-squares regression. In *Advances in Neural Information Processing Systems*, pp. 1213–1221, 2009.

Olver, S. On the convergence rate of a modified fourier series. *Mathematics of Computation*, 78(267):1629–1645, 2009.

Rakhlin, A., Sridharan, K., Tsybakov, A. B., et al. Empirical entropy, minimax regret and minimax risk. *Bernoulli*, 23 (2):789–824, 2017.

Rumerlhar, D. Learning representation by back-propagating errors. *Nature*, 323:533–536, 1986.

Schmidt-Hieber, J. Nonparametric regression using deep neural networks with relu activation function. *Annals of Statistics*, 2019.

Shalev-Shwartz, S. and Ben-David, S. *Understanding machine learning: From theory to algorithms*. 2014.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Sohl-Dickstein, J. and Kawaguchi, K. Eliminating all bad local minima from loss landscapes without even adding an extra unit. *arXiv preprint arXiv:1901.03909*, 2019.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

Telgarsky, M. Benefits of depth in neural networks. In *Conference on Learning Theory*, pp. 1517–1539, 2016.

Vershynin, R. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

Wei, C. and Ma, T. Data-dependent sample complexity of deep neural networks via lipschitz augmentation. *arXiv preprint arXiv:1905.03684*, 2019.

Wu, Y. and He, K. Group normalization. In *European Conference on Computer Vision*, pp. 3–19, 2018.

Xie, B., Liang, Y., and Song, L. Diverse neural network learns true target functions. In *International Conference on Artificial Intelligence and Statistics*, pp. 1216–1224, 2017.

Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. Aggregated residual transformations for deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5987–5995, 2017.

Xie, X., Wu, J., Zhong, Z., Liu, G., and Lin, Z. Differentiable linearized ADMM. In *International Conference on Machine Learning*, 2019.

Yarotsky, D. Error bounds for approximations with deep relu networks. *Neural Networks*, 94:103–114, 2017.

Yarotsky, D. Optimal approximation of continuous functions by very deep relu networks. In *Conference on Learning Theory*, pp. 639–649, 2018.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. In *International Conference on Machine Learning*, 2017.

Zhang, X., Ling, C., and Qi, L. The best rank-1 approximation of a symmetric tensor and related spherical optimization problems. *SIAM Journal on Matrix Analysis and Applications*, 33(3):806–821, 2012.

A. Appendix

A.1. Experimental Settings and Model Configuration

For the baseline network, it is a reduced version of the VGG network. We adopt the similar structure as that in [6], where the last layer is a fully-connected layer and all other weighted layers are convolutional layers. It contains five max pooling in total. For our MCN, we replace the convolutional layers after the third max pooling layer with our MCN block introduced in the right part of Figure 1. For each MCN block, the upper convolutional operation has 128 3×3 kernels, and other three operations has 256 3×3 kernels The model configuration of MCN is presented in Table 3. For fair comparison and for all models with different layers, we set the learning rate to 1×10^{-4} and total number of epochs to 250, respectively.

Table 3. Model configuration of MCN.

MCN Configuration					
6 weight layers	7 weight layers	8 weight layers	9 weight layers	10 weight layers	11 weight layers
Input (32×32 RGB image)					
3×3 conv. 64 BN ReLU	3×3 conv. 64 BN ReLU	3×3 conv. 64 BN ReLU	3×3 conv. 64 BN ReLU	3×3 conv. 64 BN ReLU	3×3 conv. 64 BN ReLU
					3×3 conv. 64 BN ReLU
Max pooling					
3×3 conv. 128 BN ReLU	3×3 conv. 128 BN ReLU	3×3 conv. 128 BN ReLU	3×3 conv. 128 BN ReLU	3×3 conv. 128 BN ReLU	3×3 conv. 128 BN ReLU
				3×3 conv. 128 BN ReLU	3×3 conv. 128 BN ReLU
Max pooling					
3×3 conv. 256 BN ReLU	3×3 conv. 256 BN ReLU	3×3 conv. 256 BN ReLU	3×3 conv. 256 BN ReLU	3×3 conv. 256 BN ReLU	3×3 conv. 256 BN ReLU
			3×3 conv. 256 BN ReLU	3×3 conv. 256 BN ReLU	3×3 conv. 64 BN ReLU
Max pooling					
MCN Block	MCN Block	MCN Block	MCN Block	MCN Block	MCN Block
		MCN Block	MCN Block	MCN Block	MCN Block
Max pooling					
MCN Block	MCN Block	MCN Block	MCN Block	MCN Block	MCN Block
	MCN Block	MCN Block	MCN Block	MCN Block	MCN Block
Max pooling					
FC-10					
Soft-max					

It is worth noting that our MCN contains only a small amount of parameters. Specifically, in our configuration, each MCN block has only $3 \times 3 \times (256 \times 3 + 128) = 8,064$ parameters. And MCN uses only a single fully-connected layer, which leads to $512 \times 10 = 5,120$ parameters. So, even for an MCN with 11 layers, the total number of parameters is less than 5×10^4.

A.2. Proof of Theorem 1

Proof. Since $\mathcal{L}_{k+1}, \mathcal{W}_{k+1}, \tilde{\mathcal{A}}_{k+1}$ and $\mathcal{A}_{k+1}$ are linear operators, ignoring the biases, we simplify MCN as:

$$\mathbf{x}_{k+1,i} = \left[\mathbf{L}_{k+1}\mathbf{x}_{k,i};\ \gamma\left(\tilde{\mathbf{A}}_{k+1}\mathbf{x}_i\right) + \max\left\{\mathbf{W}_{k+1}\mathbf{x}_{k,i}, \sigma\left(\mathbf{A}_{k+1}\mathbf{x}_i\right)\right\}\right],$$

where $\{\mathbf{L}_{k+1}, \mathbf{W}_{k+1}, \tilde{\mathbf{A}}_{k+1}, \mathbf{A}_{k+1}\} \in \boldsymbol{\theta}_{k+1}$, $\boldsymbol{\theta}_{k+1}$ is a local minimum of the loss L and $\mathbf{x}_i \in \{\mathbf{x}_i\}_{i=1}^n$ is an arbitrary training sample. For convenience, in this proof, we assume $\mathbf{x}_i \neq \mathbf{x}_j$ when $i \neq j$ for any $\mathbf{x}_i$ and $\mathbf{x}_j \in \{\mathbf{x}_i\}_{i=1}^n$ and $\sigma(\cdot) = \sigma_k(\cdot),\ \forall k \in [l]$.

Let $\ell_\Psi(\mathbf{x}_{k+1,i}) := \ell(\Psi(\mathbf{x}_{k+1,i}), \mathbf{y}_i)$ and $\nabla\ell_\Psi(\mathbf{x}_{k+1,i})$ be the gradient $\nabla\ell_\Psi$ evaluated at $\mathbf{x}_{k+1,i}$. We can have the following claim.

Claim 2. *With the same setting in Theorem 1, for any* $\mathbf{u} \in \mathbb{R}^{d_x}$ *with* $\|\mathbf{u}\|_2 = 1$ *and* $t \in \mathbb{N}$ *we have:*

$$\sum_{i=1}^{n} c_{i,j,t}\left(\nabla\ell_\Psi(\mathbf{x}_{k+1,i})\right)_j \left(\mathbf{u}_j{}^\top\mathbf{x}_i\right)^t = 0, \quad \forall j \in [d_{\mathcal{L}} + 1, d_{k+1}], \quad \forall i \in [n],$$

[6] https://github.com/kuangliu/pytorch-cifar/blob/master/models/vgg.py

where

$$c_{i,j,t} \coloneqq \gamma^{(t)}\left(\left(\tilde{\mathbf{A}}\mathbf{x}_i\right)_j\right).$$

Proof. Let $\mathbf{\Lambda}_{k+1,i} \in \mathbb{R}^{(d_{k+1}-d_{\mathcal{L}})\times(d_{k+1}-d_{\mathcal{L}})}$ represents a diagonal matrix with diagonal elements corresponding to the maximum pattern of the data point $\mathbf{x}_i$ at the $(k+1)$-th layer as:

$$\left(\mathbf{\Lambda}_{k+1,i}\right)_{(j,j)} \coloneqq \begin{cases} 1, & \text{if} \quad \mathbf{x}_{k,i}^\top (\mathbf{W}_{k+1}^\top)_j \leq \sigma\left(\mathbf{x}_i^\top \left(\mathbf{A}_{k+1}^\top\right)_j\right); \\ 0, & \text{otherwise,} \end{cases}$$

where $(\mathbf{A}_{k+1}^\top)_j$ is the j-th column of the matrix $\mathbf{A}_{k+1}^\top$, i.e., the j-th row of $\mathbf{A}_{k+1}$. We also define the complement of the matrix $\mathbf{\Lambda}_{k+1,i}$:

$$\overline{\mathbf{\Lambda}}_{k+1,i} = \mathbf{I} - \mathbf{\Lambda}_{k+1,i},$$

where $\mathbf{I} \in \mathbb{R}^{(d_{k+1}-d_{\mathcal{L}})}$ is the identity matrix. Without ambiguity, we omit the subscription $(k+1)$ for $\{\mathbf{L}_{k+1}, \mathbf{W}_{k+1}, \tilde{\mathbf{A}}_{k+1}, \mathbf{A}_{k+1}, \overline{\mathbf{\Lambda}}_{k+1,i}, \mathbf{\Lambda}_{k+1,i}\}$ and rewrite $\mathbf{x}_{k+1}$ as:

$$\mathbf{x}_{k+1,i} = \left[\mathbf{L}\mathbf{x}_{k,i};\ \gamma\left(\tilde{\mathbf{A}}\mathbf{x}_i\right) + \overline{\mathbf{\Lambda}}_i \mathbf{W}\mathbf{x}_{k,i} + \mathbf{\Lambda}_i\left(\sigma\left(\mathbf{A}\mathbf{x}_i\right)\right)\right].$$

By perturbing parameters, we can define a new output:

$$\mathbf{x}'_{k+1,i} = \left[\mathbf{L}\mathbf{x}_{k,i};\ \gamma\left(\left(\tilde{\mathbf{A}} + \mathbf{\Delta}\mathbf{A}\right)\mathbf{x}_i\right) + \overline{\mathbf{\Lambda}}'_i \mathbf{W}\mathbf{x}_{k,i} + \mathbf{\Lambda}'_i\left(\sigma\left(\mathbf{A}\mathbf{x}_i\right)\right)\right].$$

In general, due to the perturbation, the maximum pattern $\mathbf{\Lambda}_i$ will change. However, if the perturbation is small enough, i.e., $\|\mathbf{\Delta}\mathbf{A}\|$ is sufficiently small, we have $\mathbf{\Lambda}_i = \mathbf{\Lambda}'_i$. Then, we have:

$$\mathbf{d}_i \coloneqq x_{k+1,i} - x'_{k+1,i} = \left[\mathbf{0};\ \gamma\left(\left(\tilde{\mathbf{A}} + \mathbf{\Delta}\mathbf{A}\right)\mathbf{x}_i\right) - \gamma\left(\tilde{\mathbf{A}}\mathbf{x}_i\right)\right].$$

For any $j \in [d_{\mathcal{L}}, d_{k+1}]$, we let:

$$e_{i,j} \coloneqq \mathbf{x}_i^\top \mathbf{\Delta}\mathbf{a}_j,$$

where $\mathbf{\Delta}\mathbf{a}_j = \mathbf{\Delta}\mathbf{A}_{(j,:)}$ is the j-th row of the matrix $\mathbf{\Delta}\mathbf{A}$. Then by the Taylor expansion of the function $\gamma(\cdot)$ at $\left(\tilde{\mathbf{A}}\mathbf{x}_i\right)_j$ for all i, j, we have

$$\mathbf{d}_{i,j} = \sum_{q=1}^{\infty} \frac{\gamma^{(q)}\left(\left(\tilde{\mathbf{A}}\mathbf{x}_i\right)_j\right)}{q!} e_{i,j}^q.$$

Let $\tilde{\boldsymbol{\theta}}_{k+1} = \{\boldsymbol{\theta}_{k+1} \setminus \mathbf{A}, \mathbf{A} + \mathbf{\Delta}\mathbf{A}\}$. Since $\tilde{\boldsymbol{\theta}}_{k+1}$ is a local minimum, we have that, for any sufficiently small $\mathbf{\Delta}\mathbf{A}$, we have:

$$\begin{aligned}
&n\left(L(\boldsymbol{\theta}_{k+1}) - L(\tilde{\boldsymbol{\theta}}_{k+1})\right) = \sum_{i=1}^{n} \ell(\Psi(\mathbf{x}_{k+1,i}), \mathbf{y}_i) - \sum_{i=1}^{n} \ell(\Psi(\mathbf{x}'_{k+1,i}), \mathbf{y}_i) = \sum_{i=1}^{n} \left(\ell_\Psi(\mathbf{x}_{k+1,i}) - \ell_\Psi(\mathbf{x}'_{k+1,i})\right) \\
&\overset{(a)}{=} \sum_{i=1}^{n} \left(\left(\nabla \ell_\Psi(\mathbf{x}_{k+1,i})^\top \mathbf{d}_i + \mathcal{O}(\|\mathbf{d}_i\|^2)\right)\right) \overset{(b)}{=} \sum_{j=d_{\mathcal{L}}+1}^{d_{k+1}} \sum_{i=1}^{n} \left(\left(\nabla \ell_\Psi(\mathbf{x}_{k+1,i})\right)_j \mathbf{d}_{i,j}\right) + \mathcal{O}\left(\|\mathbf{\Delta}\mathbf{A}\|^2\right) \\
&= \sum_{j=d_{\mathcal{L}}+1}^{d_{k+1}} \left(\sum_{q=1}^{\infty} \frac{z_{j,q}}{q!}\right) + \mathcal{O}\left(\|\mathbf{\Delta}\mathbf{A}\|^2\right) \leq 0,
\end{aligned}$$

where

$$z_{j,q} \coloneqq \left(\sum_{i=1}^{n} c_{i,j,q} \left(\nabla \ell_\Psi(\mathbf{x}_{k+1,i})\right)_j e_{i,j}^q\right), \quad c_{i,j,q} \coloneqq \gamma^{(q)}\left(\left(\tilde{\mathbf{A}}\mathbf{x}_i\right)_j\right),$$

and (a) comes from the definition of differentiability for multivariable function and (b) is due to the boundness of the first derivative of $\gamma(\cdot)$. Since the sum is the dominant term, then we can have:

$$\sum_{j=d_{\mathcal{L}}+1}^{d_{k+1}} \left(\sum_{q=1}^{n} \frac{z_{j,q}}{q!} \right) \leq 0.$$

Due to this inequality holds for any sufficient small $\mathbf{\Delta a}_j$, we can conclude that

$$\sum_{q=1}^{\infty} \frac{z_{j,q}}{q!} = 0, \quad \forall j.$$

By setting $\mathbf{\Delta a}_j = \epsilon_j \mathbf{u}_j$ such that $\epsilon_j > 0$ and $\|\mathbf{u}_j\| = 1$, we have:

$$\sum_{q=1}^{\infty} \frac{\epsilon_j^q}{q!} \sum_{i=1}^{n} c_{i,j,q} \left(\nabla \ell_{\Psi}(\mathbf{x}_{k+1,i}) \right)_j \left(\mathbf{u}_j^{\top} \mathbf{x}_i \right)^q = 0, \quad \forall j.$$

Now, we set

$$\eta_q = \left(\sum_{i=1}^{n} c_{i,j,q} \left(\nabla \ell_{\Psi}(\mathbf{x}_{k+1,i}) \right)_j \left(\mathbf{u}_j^{\top} \mathbf{x}_i \right)^q \right).$$

Divide the ϵ_j on both side, we can get:

$$\eta_1 + \sum_{q=2}^{\infty} \frac{\epsilon_j^{q-1}}{q!} = 0, \quad \forall j.$$

Note that

$$\sum_{q=2}^{\infty} \frac{\epsilon_j^{q-1}}{q!} \to 0, \quad \epsilon_j 0.$$

Then, we get $\eta_1 = 0$. We can multiplying $p!/\epsilon_j^q$ on both sides and prove by induction that

$$\eta_q = 0, \text{ for } \quad q = 1, \cdots.$$

We finish the proof of this claim. □

Given any $i \in \{1, \cdots, n\}$, consider the case:

$$\left(\nabla \ell_{\Psi}(\mathbf{x}_{k+1,i}) \right)_j = 0, \quad \forall j \in [d_{\mathcal{L}}, d_{k+1}].$$

We can rewrite the above equation as:

$$\Psi^{\top} \left(\nabla \ell \left(\Psi(\mathbf{x}_{k+1,i}), y_i \right) \right) = \begin{bmatrix} * \\ \mathbf{0} \end{bmatrix},$$

where $\nabla \ell(\cdot)$ is the gradient of $\nabla \ell$, e.g., $\nabla \ell \left(\Psi(\mathbf{x}_{k+1,i}), y_i \right) = \Psi(\mathbf{x}_{k+1,i}) - y_i$ for squared loss or $\nabla \ell \left(\Psi(\mathbf{x}_{k+1,i}), y_i \right) = \eta(\Psi(\mathbf{x}_{k+1,i})) - y_i$, where $\eta(\cdot)$ is the softmax function for cross entropy loss, $\mathbf{0} \in \mathbb{R}^{d_{k+1} - d_{\mathcal{L}}}$ and $*$ is an arbitrary vector in $\mathbb{R}^{d_{\mathcal{L}}}$. Since $\Psi(\cdot)$ is surjection [7] , we can conclude that:

$$\ell \left(\Psi(\mathbf{x}_{k+1,i}), y_i \right) = 0, \quad \forall i \in [n],$$

which completes this proof. Therefore, for the sake of simplicity, we exclude this all zero case and assume $\left(\nabla \ell_{\Psi}(\mathbf{x}_{k+1,i}) \right)_{d_{\mathcal{L}}+1} \neq 0$ in the following proof.

[7] Let $\Psi(\mathbf{x}) = \begin{bmatrix} \Psi_1(\mathbf{x}) & \Psi_2(\mathbf{x}) \end{bmatrix}$. Actually, it needs $\Psi_2(\mathbf{x})$ to be surjective here. However, the entries' order of MCN's each layer can be arbitrary and $\Psi(\cdot)$ is fixed. Hence, we can always change the order of entries of $\mathbf{x}$ to let $\Psi_2(\mathbf{x})$ be surjective without changing the values of learnable parameters.

Given $\boldsymbol{\theta}_{k+1}$ is a local minimum of L, by the convexity of the function $\ell_\Psi(\mathbf{x}_{k+1,i})$, for any $\boldsymbol{\theta}'_l$, we have:

$$n\left(L(\boldsymbol{\theta}'_l) - L(\boldsymbol{\theta}_{k+1})\right) \geq \sum_{i=1}^{n} \nabla \ell_\Psi(\mathbf{x}_{k+1,i})^\top \left(\mathbf{x}'_{k,i} - \mathbf{x}_{k+1,i}\right) = \underbrace{\sum_{j=1}^{d_{k+1}} \sum_{i=1}^{n} \left(\nabla \ell_\Psi(\mathbf{x}_{k+1,i})\right)_j \left(\mathbf{x}'_{k,i} - \mathbf{x}_{k+1,i}\right)_j}_{\text{Lower Bound } L_B}.$$

Denote by $\otimes$ the tensor product and let $\mathbf{x}^{\otimes p} \coloneqq \mathbf{x} \otimes \cdots \otimes \mathbf{x}$. For a p-th order tensor $\mathbf{M} \in \mathbb{R}^{d \times \cdots \times d}$ and p vectors $\{\mathbf{u}_1, \cdots, \mathbf{u}_p\}$, let

$$\mathbf{M}\left(\mathbf{u}_1, \cdots, \mathbf{u}_p\right) \coloneqq \sum_{1 \leq i_1, \cdots, i_p \leq d} \mathbf{M}_{i_1, \cdots, i_p} \mathbf{u}_{1,i_1} \cdots \mathbf{u}_{p,i_p}.$$

It is known from (Zhang et al., 2012), given $n, p > 0$ and ξ_i for $i = 1, \cdots, n$,

$$\max_{\|\mathbf{u}_1\|_2 = \|\mathbf{u}_2\|_2 = \cdots = \|\mathbf{u}_p\|_2 = 1} \left(\sum_{i=1}^{n} \xi_i \mathbf{x}_i^{\otimes p}\right)\left(\mathbf{u}_1, \cdots, \mathbf{u}_p\right) = \max_{\|\mathbf{u}\|_2} \left(\sum_{i=1}^{n} \xi_i \left(\mathbf{u}^\top \mathbf{x}_i\right)^p\right).$$

Hence, with this observation, together with the results in Claim 2, we get

$$\sum_{i=1}^{n} c_{i,j,t} \left(\nabla \ell_\Psi(\mathbf{x}_{k+1,i})\right)_j \operatorname{vec}(\mathbf{x}_i^{\otimes t}) = 0, \quad \forall j \in [d_{\mathcal{L}} + 1, d_{k+1}], \quad \forall t \in [n]$$

Before proceeding, we provide a result of the existence of a polynomial interpolation of the finite distinct n points; interpolation of finite n points.

Claim 3 (Polynomial Interpolation(Gasca & Sauer, 2000))**.** *Let $\{\mathbf{x}_i\}_{i=1}^n$ be distinct points in $\mathbb{R}^{d_x}$. For any d_x–dimensional continuous functions $f : \mathbb{R}^{d_x} \to \mathbb{R}$, consider the set $\boldsymbol{\Omega} \coloneqq \{f(\mathbf{x}_1), \cdots, f(\mathbf{x}_n)\}$. There exists a r-th order polynomial $q(\cdot) : \mathbb{R}^{d_x} \to \mathbb{R}$ such that interpolate the points in the set $\boldsymbol{\Omega}$, where the order $r \leq (n-1)$; namely, there exists the vectors $\{\mathbf{u}_t \in \mathbb{R}^{d_x^t}\}$ for $t = 1, \cdots, r$ such that*

$$f(\mathbf{x}_i) = q(\mathbf{x}_i) = \sum_{t=1}^{r} \mathbf{u}_t^\top \operatorname{vec}\left(\mathbf{x}_i^{\otimes t}\right), \quad \forall \mathbf{x}_i \in \{\mathbf{x}_i\}_{i=1}^n.$$

By this claim, it is easy to conclude that the difference of two continuous functions $f_1(\cdot)$ and $f_2(\cdot)$ can also be interpolated; namely, there exists vectors $\{\mathbf{u}_t^{(1)} \in \mathbb{R}^{d_x^t}\}$ and $\{\mathbf{u}_t^{(2)} \in \mathbb{R}^{d_x^t}\}$:

$$f_1(\mathbf{x}_i) - f_2(\mathbf{x}_i) = \sum_{t=1}^{r} {\mathbf{u}_t^{(1)}}^\top \operatorname{vec}\left(\mathbf{x}_i^{\otimes t}\right) - \sum_{t=1}^{r} {\mathbf{u}_t^{(2)}}^\top \operatorname{vec}\left(\mathbf{x}_i^{\otimes t}\right) = \sum_{t=1}^{r} \left(\mathbf{u}_t^{(1)} - \mathbf{u}_t^{(2)}\right)^\top \operatorname{vec}\left(\mathbf{x}_i^{\otimes t}\right) \coloneqq \sum_{t=1}^{r} \mathbf{u}_t^\top \operatorname{vec}\left(\mathbf{x}_i^{\otimes t}\right).$$

Note that when $\gamma(\cdot) = \exp(\cdot)$, we have $c_{i,j,t_1} = c_{i,j,t_2}$ when $t_1 \neq t_2$. Hence, we omit the subscript t. Notice that for any $j \in [d_{k+1}]$, $\left(\mathbf{x}'_{k,i}\right)_j$ and $\left(\mathbf{x}_{k+1,i}\right)_j$ are always continuous functions of $\mathbf{x}_i$. Hence, for all i, there exists vectors $\{\mathbf{u}_{t,j} \in \mathbb{R}^{d_x^t}\}$ such that:

$$\frac{1}{c_{i,j}} \left(\mathbf{x}'_{k,i} - \mathbf{x}_{k+1,i}\right)_j \frac{\left(\nabla \ell_\Psi(\mathbf{x}_{k+1,i})\right)_j}{\left(\nabla \ell_\Psi(\mathbf{x}_{k+1,i})\right)_{d_{\mathcal{L}}+1}} = \sum_{t=1}^{r} \mathbf{u}_{t,j}^\top \operatorname{vec}\left(\mathbf{x}_i^{\otimes t}\right), \quad \forall j \in [d_{\mathcal{L}}], \tag{5}$$

and

$$\frac{1}{c_{i,j}} \left(\mathbf{x}'_{k,i} - \mathbf{x}_{k+1,i}\right)_j = \sum_{t=1}^{r} \mathbf{u}_{t,j}^\top \operatorname{vec}\left(\mathbf{x}_i^{\otimes t}\right), \quad \forall j \in [d_{\mathcal{L}} + 1, d_{k+1}]. \tag{6}$$

If $\left(\nabla \ell_\Psi(\mathbf{x}_{k+1,i})\right)_j = 0$ for some $j \in [d_{\mathcal{L}}]$, then we can ignore this zero term in the lower bound L_B. Thus, for brevity, we assume that $\left(\nabla \ell_\Psi(\mathbf{x}_{k+1,i})\right)_j \neq 0$, $\forall j \in [d_{\mathcal{L}}]$. Combing the Eq. (5) and Eq. (6), we have

$$\begin{aligned}
L_B &= \sum_{t=1}^{r} \sum_{j=1}^{d_{\mathcal{L}}} \sum_{i=1}^{n} c_{i,j} \left(\nabla \ell_\Psi(\mathbf{x}_{k+1,i})\right)_{d_{\mathcal{L}}+1} \mathbf{u}_{t,j}^\top \operatorname{vec}\left(\mathbf{x}_i^{\otimes t}\right) + \sum_{t=1}^{r} \sum_{j=d_{\mathcal{L}}+1}^{d_{k+1}} \sum_{i=1}^{n} c_{i,j} \left(\nabla \ell_\Psi(\mathbf{x}_{k+1,i})\right)_j \mathbf{u}_{t,j}^\top \operatorname{vec}\left(\mathbf{x}_i^{\otimes t}\right) \\
&= \sum_{t=1}^{r} \sum_{j=1}^{d_{\mathcal{L}}} \mathbf{u}_{t,j}^\top \left(c_{i,j} \sum_{i=1}^{n} \left(\nabla \ell_\Psi(\mathbf{x}_{k+1,i})\right)_{d_{\mathcal{L}}+1} \operatorname{vec}\left(\mathbf{x}_i^{\otimes t}\right)\right) + \sum_{t=1}^{r} \sum_{j=d_{\mathcal{L}}+1}^{d_{k+1}} \mathbf{u}_{t,j}^\top \left(\sum_{i=1}^{n} \left(\nabla \ell_\Psi(\mathbf{x}_{k+1,i})\right)_j \operatorname{vec}\left(\mathbf{x}_i^{\otimes t}\right)\right) \\
&= 0,
\end{aligned}$$

where the last equality comes from the Claim 2. Therefore, when $\boldsymbol{\theta}_{k+1}$ is a local minimum of L, we have $L(\boldsymbol{\theta}_l') \geq L(\boldsymbol{\theta}_{k+1})$ for any $\boldsymbol{\theta}_l'$.

We now complete this proof. □

A.3. Proof of Theorem 2

Proof. We first provide several claims. Based on them, we can construct an MCN such that approximate the multivariate Fourier series will, which ensure the accurateness for approximation in the Sobolev space.

Claim 4. *The function $f(x) = x^2$ on the segment $[-1, 1]$ can be approximated by an MCN of width $\mathcal{O}(w)$ and depth $\mathcal{O}(l)$ with the approximation error:*

$$\epsilon = \mathcal{O}(2^{-wl}).$$

When l is large enough, the number of non-zero parameters for this MCN is in the order of $\mathcal{O}(w^2 l)$.

Proof. We only consider the proof on the interval $[0, 1]$, the other half is the same. Consider the $g : [0, 1] \to [0, 1]$,

$$g_m(x) \coloneqq \max\{-\frac{x}{2}, \frac{x}{2} - 2^{1-2m}\},$$

and the nested function

$$r_m(x) = g_m \circ g_{m-1} \circ \cdots \circ g_1(x).$$

It is easy to see that $r_m(x)$ can be represented by the operator $\mathcal{M}(\cdot)$ in MCN (see Eq. (2)). Hence, we can have one type of MCN $M_i(x)$ such that

$$M_i \coloneqq \begin{bmatrix} \sum_i \\ g_{i+1} \\ g_{i+2} \\ \vdots \\ g_{i+m} \end{bmatrix}, \text{ then } M_i \circ \left(\begin{bmatrix} r_i \\ r_{i+1} \\ \vdots \\ r_{i+m-1} \end{bmatrix} \right) = \begin{bmatrix} \sum_i^{i+m} r_i \\ r_{i+1} \\ r_{i+2} \\ \vdots \\ r_{i+m}. \end{bmatrix}.$$

Now, we construct a three-layer MCN with m units $[r_1, \cdots, r_m()]$ as the output. Note that

$$r_m(x) = \begin{cases} 2^{-m}\left(\frac{2k}{2^m} - x\right), & x \in \left[\frac{2k}{2^m}, \frac{2k+1}{2^m}\right], k = 0, 1, \cdots, 2^{m-1} - 1, \\ 2^{-m}\left(x - \frac{2k}{2^m}\right), & x \in \left[\frac{2k-1}{2^m}, \frac{2k}{2^m}\right], k = 1, 2, \cdots, 2^{m-1}. \end{cases}$$

is a "sawtooth" function. We now let $\mathcal{A}_1(x)$ and $\mathcal{W}_1(x)$ be:

$$\mathcal{A}_1(x) = \begin{bmatrix} 2^{-1}(-x) \\ \vdots \\ 2^{-s}\left(\frac{2k_s}{2^s} - x\right) \end{bmatrix}, \mathcal{W}_1(x) = \begin{bmatrix} 2^{-1}(x-1) \\ \vdots \\ 2^{-s}\left(x - \frac{2k_s'}{2^s}\right) \end{bmatrix}, \quad s = 1, \cdots, m, \quad k_s' - 1 = k_s = 0, 1, \cdots, 2^{s-1} - 1.$$

Hece, $\mathcal{A}_1$ and $\mathcal{W}_1$ map the input from $\mathbb{R} \to \mathbb{R}^p$, where $p = 2^m - 1$ and the $(2^{s-1} + k_s)$-th entry of $\mathcal{M}_1(x) = \max\{\mathcal{A}_1(x), \mathcal{W}_1(x)\}$ is the k_s-th "tooth" of $r_s(x)$ when $r_s(x) < 0$. Let $\mathcal{W}_2$ be the sign reversal operator and $\mathcal{A}_2$ be the zero mapping, then we have

$$\mathcal{M}_2(x) = \max\{-\mathcal{M}_1(x), 0\} = \begin{cases} -r_s(x), & x \in \left[\frac{2k_s-1}{2^m}, \frac{2k_s+1}{2^m}\right], \\ 0, & \text{otherwise.} \end{cases}$$

At last we let $\mathcal{A}_3(\cdot) = -1$ and

$$\mathcal{W}_3(x) = -\begin{bmatrix} \sum_{i=1}^2 x_i \\ \vdots \\ \sum_{i=2^{m-1}}^{2^m} x_i \end{bmatrix}, \quad \text{then } \mathcal{M}_3(x) = \max\{-1, \mathcal{M}_2(x)\} = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_m \end{bmatrix}.$$

We define the above three layer MCN as $M_0(x) \coloneqq \mathcal{M}_3(x)$. Then we have a $(l+3)$-layer MCN $M(x)$ such that

$$M(x) \coloneqq M_l \circ M_{l-1} \circ \cdots \circ M_0(x), \quad \text{with } \tilde{\mathcal{A}}_l(x) = x \text{ and } \tilde{\mathcal{A}}_k(x) = 0, \forall k \leq l.$$

It is obvious that the first entry of $M(x) = x + \sum_{i=1}^{ml} r_i(x)$. Form the previous results, e.g., Proposition 2 in (Yarotsky, 2017) and Lemma A.1. in (Schmidt-Hieber, 2019), we already have

$$\left| x + \sum_{i=1}^{ml} r_i(x) - x^2 \right| \leq 2^{-ml} \leq 2^{-wl}$$

We can easily find that the number of the non-zero parameters for $M(x)$ is in the order of $O(w^2 l + 2^m)$. However, since MCN has the concatenation operator as in the Eq. (2), we can expand the width of M_i so that the $w = \dim(M_i) >> m$, when l is large, we can have $w^2 l \geq 2^m$; and finish the proof. □

Note that

$$xy = \frac{1}{2}\left((x+y)^2 - x^2 - y^2\right),$$

we can use Claim 4 to efficiently approximate polynomial by MCN.

Claim 5. *The function $f(\mathbf{x}) = \prod_{i=1}^{p} x_i$ on $[-1,1]^p$ can be approximated by an MCN $\tilde{M}_p(\mathbf{x})$ of width $\mathcal{O}(wp)$ and depth $\mathcal{O}(l \ln p)$, with the error bound as:*

$$\left| \tilde{M}_p(\mathbf{x}) - \prod_{i=1}^{p} x_i \right| \leq \mathcal{O}(p 2^{-wl}).$$

The number of non-zero parameters for this MCN is in the order of $\mathcal{O}(pw^2 l)$.

Proof. We already have a $(l+3)$-layer MCN $M(x)$ such that can approximate x^2 accurately. We can easily get a $(l+5)$-layer modified MCN $\tilde{M}(x,y)$ such that $\tilde{M}(x,y) \approx xy$. $\tilde{M}$ can be obtained by

$$\tilde{M}(x,y) \coloneqq \begin{bmatrix} x \\ y \end{bmatrix} \to \begin{bmatrix} x+y \\ x \\ y \end{bmatrix} \to \begin{bmatrix} M(x+y) \\ M(x) \\ M(y) \end{bmatrix} \to \frac{1}{2}\left(M(x+y) - M(x) - M(y)\right),$$

It is obvious that

$$|\tilde{M}(x,y) - xy| \leq \frac{3}{2} \cdot 2^{-wl} = \mathcal{O}(2^{-wl}) \coloneqq \epsilon,$$

and the number of non-zero parameters for $\tilde{M}(x,y)$ is also in the order of $O(w^2 l)$. Based on the above observation, we can construct an MCN such that approximate $\prod_{i=1}^{p} x_i$. Denote $i \coloneqq \lceil \log_2(p) \rceil$. In the first layer, we computer

$$\mathbf{x} \to \left[x_1, \cdots, x_p, \underbrace{1, \cdots, 1}_{2^i - q} \right]^\top \coloneqq \mathbf{y},$$

then we define the multivariate version $\tilde{M}(\mathbf{y})$ for $\mathbf{y} \in \mathbb{R}^{2^j}$, whre $j \in \mathbb{N}_+$,

$$\tilde{M}(\mathbf{y}) \coloneqq \mathbf{y} \to \begin{bmatrix} \tilde{M}(y_1, y_2) \\ \vdots \\ \tilde{M}(y_{2^j}, y_{2^j - 1}) \end{bmatrix}.$$

Then we can have a $(li + 5i + 1)$-layer MCN $\tilde{M}_p(\mathbf{x})$, with the width be wp, such that,

$$\tilde{M}_p(\mathbf{x}) \coloneqq \underbrace{\tilde{M} \circ \cdots \circ \tilde{M}(\mathbf{y})}_{i}.$$

Note that, for $a, b, c, d \in [-1, 1]$, we have

$$\tilde{M}(a, b) - cd \leq \epsilon + |a - c| + |b - d|.$$

Recall that $i \coloneqq \lceil \log_2(p) \rceil$ and omit the high order terms of ϵ, we get

$$\left| \tilde{M}_p(\mathbf{x}) - \prod_{i=1}^{p} x_i \right| \leq \sum_{k=0}^{i-1} 2^k \epsilon \leq 2^i \epsilon = \mathcal{O}(p 2^{-wl}).$$

It is easy to verify that the number of non-zero parameters is in the order of $\mathcal{O}(2^i + p \cdot w^2 l) = \mathcal{O}(pw^2 l)$. □

Claim 6. *The function $f(x) = \sum_{j=1}^{p} a_j x^j$, where and $x \in [-1, 1]$, can be approximated by MCN M_{poly} of width $\mathcal{O}(wp \ln p)$ and depth $\mathcal{O}(l \ln p)$, with the error bound as:*

$$\left| \tilde{M}_{\text{poly}}(x) - \sum_{j=1}^{p} a_j x^j \right| \leq \mathcal{O}(\|\mathbf{a}\|_1 p^2 2^{-wl}).$$

The number of non-zero parameters for this MCN is in the order of $\mathcal{O}(w^2 lp \ln p)$.

Proof. We first the copy x p-times

$$\mathbf{x}_p \coloneqq [\underbrace{x, \cdots, x}_{p}]^\top.$$

We then apply the MCN $\tilde{M}_p(\mathbf{x}_p)$ in Claim 5 to it to approximate x^p. Interestingly, since MCN has the skip-connection with any previous layer by the operator $\mathcal{A}_k()$, hence from the MCN $\tilde{M}_p(\mathbf{x}_p)$ in Claim 5 we can extract

$$\mathbf{y} \coloneqq \left[\tilde{M}_1(\mathbf{x}_p), \tilde{M}_2(\mathbf{x}_p), \tilde{M}_4(\mathbf{x}_p), \cdots, \tilde{M}_{2^i}(\mathbf{x}_p) \right] \approx \left[x, x^2, x^4, \cdots, x^{2^i} \right],$$

where $i \coloneqq \lceil \log_2(p) \rceil$. We now append p sub-MCNs on $\mathbf{y}$ to approximate x^j for $j = 1, \cdots, p$. Each sub-MCN first need to choose components from $\mathbf{y}$, then use the MCN $\tilde{M}_p(\cdot)$ in Claim 5 to "multiply" the components, e.g.,

$$x^7 = x \cdot x^2 \cdot x^4 \approx \tilde{M}_3 \left(\begin{bmatrix} \tilde{M}_1(\mathbf{x}_p) \\ \tilde{M}_2(\mathbf{x}_p) \\ \tilde{M}_4(\mathbf{x}_p) \end{bmatrix} \right).$$

By the property of telescoping sum and the results in the previous Claim, the approximation error for x^7 is in the order $\mathcal{O}(3 \cdot 2^{-wl} + 3 \cdot 2^{-wl})$. Actually, finding such a sub-MCN for x^j is equivalent to expressing j in binary. Hence, the approximation error for each sub-MCN $\tilde{M}_j^{\text{sub}}$ which aims at x^j is

$$\left| \tilde{M}_j^{\text{sub}}(\mathbf{y}) - x^j \right| \leq \mathcal{O}\left(\ln p \cdot 2^{-wl} + (\sum_{k=0}^{i} 2^k) \cdot 2^{-wl} \right) = \mathcal{O}(p 2^{-wl}).$$

Therefore, Let

$$\tilde{M}_{\text{poly}}(\mathbf{x}) \coloneqq \sum_{j=1}^{p} a_j \tilde{M}_j^{\text{sub}}(\mathbf{y}),$$

then

$$\left| \tilde{M}_{\text{poly}}(\mathbf{x}) - \sum_{j=1}^{p} a_j x^j \right| \leq \mathcal{O}(\|\mathbf{a}\|_1 p^2 2^{-wl}).$$

The total number of non-zero parameters for $\tilde{M}_j^{\text{sub}}(\mathbf{y})$ is in the order of

$$\mathcal{O}\left(\sum_{k=1}^{i} \binom{i}{k} kw^2 l \right) = \mathcal{O}(p \ln p w^2 l).$$

Hence, by adding the parameters in MCN $\tilde{M}_p(\mathbf{x}_p)$ which maps $\mathbf{x}_p$ to $\mathbf{y}$, the non-zero parameters of $\tilde{M}_{\text{poly}}(\mathbf{x})$ is in the order

$$\mathcal{O}(p \ln p w^2 l + 2p + \ln p + p w^2 l) = \mathcal{O}(w^2 l p \ln p).$$

□

Claim 7. *The function $f(\mathbf{x}) = \cos(n\pi x)$ or $f(\mathbf{x}) = \sin\left((n - \frac{1}{2})\pi x\right)$, where $n \in \mathbb{N}_+ = \mathbb{N}\backslash\{0\}$ and $x \in [-1, 1]$, can be approximated by MCNs $M_{\cos}$ and $M_{\sin}$ of width $\mathcal{O}(wp \ln p)$ and depth $\mathcal{O}(l \ln p + n^2)$, with the proper activation function and the error bound is:*

$$\epsilon = \mathcal{O}(p^{-p} \exp(p) + p^2 2^{-wl}).$$

The number of non-zero parameters for this MCN is in the order of $\mathcal{O}\left(w^2 l p \ln p + n^2\right)$.

Proof. We first consider the case $n = 1$ for $\cos(n\pi x)$. Let $y \coloneqq \pi x$, then $y \in [-\pi, \pi]$. We now need to construct an MCN to approximate $\cos(y)$ on the interval $[-\pi, \pi]$. First, we can divide the interval $[-\pi, \pi]$ into several sub-intervals and each sub-interval has the length smaller than 1, e.g., $[0, \pi/4]$ and $[\pi/4, \pi/2]$. Then we perform the Taylor expansion on each sub-interval, say $[0, \pi/4]$ for example. Since the derivative of $\cos(y)$ up to any order is bounded, the proof for other sub-interval share a similar roadmap. Note that

$$\cos(y) = \sum_{n=0}^{\infty} (-1)^n \frac{y^{2n}}{(2n)!},$$

Hence, when the even number p is large, we have

$$\left| \cos(y) - \sum_{n=0}^{p} (-1)^n \frac{y^{2n}}{(2n)!} \right| \leq \mathcal{O}(\frac{|y|^p}{p!}) \leq \mathcal{O}(\frac{1}{p!}) = \mathcal{O}\left(p^{-p-\frac{1}{2}} \exp(p)\right),$$

where the last equality comes from the Stirling's formula. Based on the results in Claim 6, there exists an MCN $M_{\cos}^{n=1}$ such that

$$\tilde{M}_{\cos}^{n=1} \approx \sum_{n=0}^{p} (-1)^n \frac{y^{2n}}{(2n)!}, \quad \forall x \in [0, \pi],$$

with the approximation error in the order $\mathcal{O}(p^2 2^{-wl} \exp(1))$, hence we parallelize all the MCNs $\tilde{M}_{\cos}^{n=1}$ on each sub-interval and obtain a final MCN $M_{\cos}^{n=1}$ of width $O(wp \ln p)$ and depth $O(l \ln p)$ such that

$$\left| M_{\cos}^{n=1}(x) - \cos(\pi x) \right| \leq \mathcal{O}(p^{-p} \exp(p) + p^2 2^{-wl}) \coloneqq \epsilon.$$

By the periodicity of $\cos(x)$, we have

$$\cos(n\pi x) = \cos\left(n\pi x - \lfloor \frac{x n^2}{2} \rfloor \frac{2\pi}{n} \right),$$

where $\lfloor \cdot \rfloor$ is the floor operator. We now need to construct an MCN which can exact perform the floor operator. Actually this can be easily implemented by choosing proper activation. Let the activation be the binary step function:

$$\sigma(x) = \begin{cases} 0, & \text{for } x < 0, \\ 1, & \text{for } x \geq 0. \end{cases}$$

Then we can obtain the floor operator on the interval $[0, n^2/2]$ by an MCN M_f of width $O(1)$ and depth $O(n^2)$

$$M_f(x) = \sum_{j=1}^{\lfloor \frac{n^2}{2} \rfloor} \sigma(x - j),$$

By the oddness of the floor operator, we can obtain $\lfloor y \rfloor$ for $y \in [-n^2/2, 0]$ without adding the depth. Hence, we can have an MCN $M_{\cos}^n$ of width $O(wp \ln p)$ and depth $O(l \ln p + n^2)$ such that

$$|M_{\cos}^n(x) - \cos(n\pi x)| \leq \epsilon, \quad \forall x \in [1, 1].$$

It is obvious that the number of the non-zero parameters of $M_{\cos}^{n}(x)$ is in the order

$$\mathcal{O}\left(w^2 lp \ln p + n^2\right).$$

Note that we can get the approximation of $\cos(k\pi x)$ for all $k = 1, \cdots, n$ from the intermediate layers of $M_{\cos}^{n}(x)$ without recalculation. Recall the definition of the Dirichlet kernel, we have

$$1 + 2\cos x + 2\cos 2x + 2\cos 3x + \cdots + 2\cos(nx) = \frac{\sin\left[\left(n + \frac{1}{2}\right)x\right]}{\sin\frac{x}{2}},$$

Hence, we can easily obtain the approximation of $\sin\left((n - \frac{1}{2})\pi x\right)$ based on the intermediate layers of MCN $M_{\cos}^{n}(x)$ without add the size of network. □

Now let

$$\phi_0^{[0]}(x) = \frac{1}{\sqrt{2}}, \quad \phi_n^{[0]}(x) = \cos(n\pi x), \quad \phi_n^{[1]}(x) = \sin\left(\left(n - \frac{1}{2}\right)\pi x\right),$$

where

$$n \in \mathbb{N}_+, \quad x \in [-1, 1].$$

Given multi-indices $\mathbf{n} = (n_1, \cdots, n_d) \in \mathbb{N}^d$ and $\mathbf{i} = (i_1, \cdots, i_d) \in \{0, 1\}^d$, we define a d-variate functions

$$\phi_{\mathbf{n}}^{[\mathbf{i}]}(\mathbf{x}) = \prod_{j=1}^{d} \phi_{n_j}^{[i_j]}(x_j), \quad \mathbf{x} = (x_1, \ldots, x_d) \in [-1, 1]^d.$$

From a standard result of spectral theory, the set $\left\{\phi_{\mathbf{n}}^{[\mathbf{i}]} : \mathbf{n} \in \mathbb{N}^d, \mathbf{i} \in \{0, 1\}^d\right\}$ is an orthonormal basis of $\mathrm{L}^2(-1, 1)^d$. We can also construct MCNs which approximate $\phi_{\mathbf{n}}^{[\mathbf{i}]}(\mathbf{x})$ well.

Claim 8. *The function $\phi_{\mathbf{n}}^{[\mathbf{i}]}(\mathbf{x})$ can be approximated by MCNs M_ϕ of width $\mathcal{O}(dwp \ln p)$ and depth $\mathcal{O}(l \ln p + \|\mathbf{n}\|_\infty^2)$, with the error bound as:*

$$\left|M_\phi(\mathbf{x}) - \phi_{\mathbf{n}}^{[\mathbf{i}]}(\mathbf{x})\right| = \mathcal{O}\left(d\left(p^{-p}\exp(p) + p^2 2^{-wl}\right)\right).$$

The number of non-zero parameters for this MCN is in the order of $\mathcal{O}\left(dw^2 p \ln p + \|\mathbf{n}\|_2^2\right)$.

Proof. For each entry of the vector $\mathbf{x}$, we append the MCNs $M_{\cos}^{(n=n_j)}$ or $M_{\sin}^{(n=n_j)}$ form the Claim 7 to approximate the function $\phi_{n_j}^{[i_j]}(x_j)$. Then, we "multiply" the functions $\phi_{n_j}^{[i_j]}(x_j)$ at the last layer by the MCN in Claim 5, hence the approximation error is

$$\mathcal{O}\left(d\left(p^{-p}\exp(p) + p^2 2^{-wl}\right) + p2^{-wl}\right) = \mathcal{O}\left(d\left(p^{-p}\exp(p) + p^2 2^{-wl}\right)\right),$$

while $M_\phi(\mathbf{x})$ is in the width $\mathcal{O}(dwp \ln p)$ and depth $\mathcal{O}(l \ln p + \|\mathbf{n}\|_\infty^2)$. We sum all the parameters in the $M_{\cos}^{(n=n_j)}$ or $M_{\sin}^{(n=n_j)}$, the non-zero parameters for $M_\phi(\mathbf{x})$ is in the order of

$$\mathcal{O}\left(dw^2 p \ln p + \|\mathbf{n}\|_2^2\right).$$

□

Claim 8 shows that there exists MCNs M_ϕ such can approximate the orthonormal basis of $\mathrm{L}^2(-1, 1)^d$ well.

For a function $\mathbf{f} \in \mathrm{L}^2(-1, 1)^d$, a truncation parameter $N \in \mathbb{N}$ and finite index set $I_N \in \mathbb{N}^d$, we can get the truncated Fourier series of $\mathbf{f}$

$$\mathcal{F}_N[\mathbf{f}](\mathbf{x}) = \sum_{\mathbf{i} \in [0,1]^d,\ \mathbf{n} \in I_N} \hat{\mathbf{f}}_{\mathbf{n}}^{[\mathbf{i}]} \phi_{\mathbf{n}}^{[\mathbf{i}]}(\mathbf{x}), \quad \text{where} \quad \hat{\mathbf{f}}_{\mathbf{n}}^{[\mathbf{i}]} = \int_{(-1,1)^d} \mathbf{f}(\mathbf{x}) \phi_{\mathbf{n}}^{[\mathbf{i}]}(\mathbf{x}) \mathrm{d}\mathbf{x}.$$

Before preceding, we provide a previous result to bound the Fourier coefficients.

Lemma 6. *Suppose that* $\mathbf{f}$ *satisfy the Condition 1. Then*

$$\left|\hat{\mathbf{f}}_{\mathbf{n}}^{[\mathbf{i}]}\right| \leq C(\chi(n), d, k) \left(\bar{n}_1 \cdots \bar{n}_d\right)^{-2(s+1)} \|f\|_{2s+2,\mathcal{H}}, \quad \mathbf{n} \in \mathbb{N}^d$$

where $\bar{m} = \max\{m, 1\}$ *for* $m \in \mathbb{N}$, $C(\chi(n), d, k)$ *is a constant only depends on the* $\chi(n)$ *(the number of non-zero entries in* $\mathbf{n}$*), dimension* d *and the smoothness of* $\mathbf{f}$*; and*

$$\|\mathbf{f}\|_{s,\mathcal{H}}^2 = \sum_{\|\boldsymbol{\alpha}\|_\infty \leq s} \left\|\mathrm{D}^{\boldsymbol{\alpha}}\mathbf{f}\right\|^2,$$

Proof. The proof can be found in (Olver, 2009) and Theorem 2.14 in (Adcock, 2010). □

We now suppose that $N = 2^r$ and let

$$I_N = \bigcup_{\|\boldsymbol{\alpha}\|_1 \leq r} \rho(\boldsymbol{\alpha}),$$

where

$$\rho(\boldsymbol{\alpha}) = \left\{\mathbf{n} \in \mathbb{N}^d : \left\lfloor 2^{\alpha_j - 1} \right\rfloor \leq n_j < 2^{\alpha_j}, \;\; j = 1, \cdots, d\right\}, \quad \boldsymbol{\alpha} \in \mathbb{N}^d.$$

We consider the size of I_N in the following lemma.

Lemma 7. *The number of terms in the set* I_N *is*

$$\frac{N(\ln N)^{d-1}}{(d-1)!} + \mathcal{O}\left(N(\ln N)^{d-2}\right).$$

Proof. The proof for the size of I_N can be found in (Huybrechs et al., 2011). □

We now provide the asymptotic order of $\mathcal{F}_{\boldsymbol{\alpha}}[\mathbf{f}](\mathbf{x})$.

Lemma 8. *Suppose that* $\mathbf{f}$ *satisfy the Condition 1. Let*

$$\mathcal{F}_{\boldsymbol{\alpha}}[\mathbf{f}](\mathbf{x}) = \sum_{\mathbf{i} \in \{0,1\}^d} \sum_{\mathbf{n} \in \rho(\boldsymbol{\alpha})} \hat{\mathbf{f}}_{\mathbf{n}}^{[\mathbf{i}]} \phi_{\mathbf{n}}^{[\mathbf{i}]}(\mathbf{x}), \quad \boldsymbol{\alpha} \in \mathbb{N}^d.$$

Then we have

$$\mathcal{F}_{\boldsymbol{\alpha}}[\mathbf{f}](\mathbf{x}) = \mathcal{O}\left(2^{-2(s+1)\|\boldsymbol{\alpha}\|_1}\right), \quad \|\boldsymbol{\alpha}\|_1 \to \infty.$$

Proof. The proof for the asymptotic order of $\mathcal{F}_{\boldsymbol{\alpha}}[\mathbf{f}](\mathbf{x})$ refers to the Eq. (4.8) in (Adcock, 2010). □

Now all the things are ready, we first consider the reminder of $\mathcal{F}_N[\mathbf{f}](\mathbf{x})$

$$\begin{aligned}|\mathbf{f} - \mathcal{F}_N[\mathbf{f}](\mathbf{x})| &= \sum_{\|\boldsymbol{\alpha}\|_1 > r} \mathcal{F}_{\boldsymbol{\alpha}}[\mathbf{f}](\mathbf{x}) = \mathcal{O}\left(\sum_{\|\boldsymbol{\alpha}\|_1 > r} \left(2^{-2(s+1)\|\boldsymbol{\alpha}\|_1}\right)\right) \\ &= \mathcal{O}\left(\int_{\|\boldsymbol{\alpha}\|_1 > r} 2^{-2(s+1)\|\boldsymbol{\alpha}\|_1}\right).\end{aligned}$$

Let $t_1^2 = |\alpha_1|, t_2^2 = |\alpha_2|, \cdots, t_d^2 = |\alpha_d|$, then we have

$$\begin{aligned}
&\int_{\|\mathbf{t}\|_2^2 > r} 2^{-2(s+1)\|\mathbf{t}\|_2^2} \\
=&\int_{\varphi_{d-1}=0}^{2\pi} \int_{\varphi_{d-2}=0}^{\pi} \cdots \int_{\varphi_1=0}^{\pi} \int_{\tilde{r}=\sqrt{r}}^{\infty} 2^d \prod_{i=1}^{d} t_i \cdot 2^{-2(s+1)\tilde{r}^2} \tilde{r}^{d-1} \sin^{d-2}(\varphi_1) \sin^{d-3}(\varphi_2) \cdots \sin(\varphi_{d-2}) \, d\tilde{r} d\varphi_1 \cdots d\varphi_{d-1} \\
=&2^d \int_0^{2\pi} \cdots \int_0^{\pi} \int_{\sqrt{r}}^{\infty} 2^{-2(s+1)\tilde{r}^2} \tilde{r}^{2d-1} \prod_{i=1}^{d} \cos(\varphi_i) \sin^{2d-3}(\varphi_1) \sin^{2d-5}(\varphi_2) \cdots \sin^3(\varphi_{d-2}) \sin(\varphi_{d-1}) \, d\tilde{r} d\varphi_1 \cdots d\varphi_{d-1} \\
=&\mathcal{O}\left(\int_{\sqrt{r}}^{\infty} 2^{-2(s+1)\tilde{r}^2} \tilde{r}^{2d-1} d\tilde{r} \right) = \mathcal{O}\left(\int_r^{\infty} 2^{-2(s+1)u} u^{d-1} du \right) = \mathcal{O}\left(2^{-2(s+1)r} r^{d-1} \right).
\end{aligned}$$

Hence, we can get

$$|\mathbf{f} - \mathcal{F}_N[\mathbf{f}](\mathbf{x})| = \mathcal{O}\left(2^{-2(s+1)r} r^{d-1} \right) = \mathcal{O}\left(N^{-2s-2} (\ln N)^{d-1} \right).$$

We then consider the approximation error for $\mathcal{F}_N[\mathbf{f}](\mathbf{x})$ by MCN. By Lemma 6, we know that

$$\left| \hat{\mathbf{f}}_{\mathbf{n}}^{[\mathbf{i}]} \right| = \mathcal{O}\left(\bar{n}_1 \cdots \bar{n}_d \right)^{-2(s+1)}.$$

Similar to the proof of the reminder term, we use the power of 2 to represent $\bar{n}_i = 2^{\alpha_i}$ for $i = 1, \cdots, d$, then

$$\sum_{\mathbf{i} \in [0,1]^d, \, \mathbf{n} \in I_N} \hat{\mathbf{f}}_{\mathbf{n}}^{[\mathbf{i}]} \leq \sum_{\mathbf{i} \in [0,1]^d, \, \mathbf{n} \in I_N} |\hat{\mathbf{f}}_{\mathbf{n}}^{[\mathbf{i}]}| \leq 2^d \sum_{\|\boldsymbol{\alpha}\|_1 \leq r} 2^{-2(s+1)\|\boldsymbol{\alpha}\|_1} \leq \mathcal{O}\left(2^d \int_{\|\boldsymbol{\alpha}\|_1 \leq r} 2^{-2(s+1)\|\boldsymbol{\alpha}\|_1} \right).$$

By a similar calculation above, we can get

$$\mathcal{O}\left(2^d \int_{\|\boldsymbol{\alpha}\|_1 \leq r} 2^{-2(s+1)\|\boldsymbol{\alpha}\|_1} \right) = \mathcal{O}\left(2^d \int_0^r 2^{-2(s+1)u} u^{d-1} du \right) = \mathcal{O}\left(2^d \right).$$

Note that, when p is large, there exists MCNs $M_\phi(\mathbf{x})$ such that

$$\left| M_\phi(\mathbf{x}) - \phi_{\mathbf{n}}^{[\mathbf{i}]}(\mathbf{x}) \right| = \mathcal{O}\left(d\left(p^{-p} \exp(p) + p^2 2^{-wl} \right) \right) = \mathcal{O}\left(dp^2 2^{-wl} \right) \coloneqq \epsilon.$$

Hence we combine all the MCNs $M_\phi(\mathbf{x})$ together to get an MCN $M_{\mathcal{F}_N}$ such that

$$|M_{\mathcal{F}_N} - \mathcal{F}_N[\mathbf{f}](\mathbf{x})| \leq \sum_{\mathbf{i} \in [0,1]^d, \, \mathbf{n} \in I_N} |\hat{\mathbf{f}}_{\mathbf{n}}^{[\mathbf{i}]}| \epsilon = \mathcal{O}\left(d 2^d p^2 2^{-wl} \right).$$

For any $\mathbf{n}$ with strictly positive entries there are 2^d choices of $\mathbf{i} \in \{0,1\}^d$. The total number of coefficients $\hat{\mathbf{f}}_{\mathbf{n}}^{[\mathbf{i}]}$ where at least one entry of n is zero is $\mathcal{O}(N(\ln N)^{d-1}$ by Lemma 7. Hence the total number of the coefficient in $\mathcal{F}_N[\mathbf{f}](\mathbf{x})$ is in the order of

$$\frac{2^d}{(d-1)!} N (\ln N)^{d-1} + \mathcal{O}\left(N (\ln N)^{d-2} \right).$$

When d is large, by the Stirling's formula, we have

$$\frac{2^d}{(d-1)!} \to 0, \quad \text{as } d \to \infty.$$

Hence, we have $\mathcal{O}\left(N(\ln N)^{d-2} \right)$ MCNs $M_\phi(\mathbf{x})$ to combine. The MCN $M_{\mathcal{F}_N}$ is in the width of $\mathcal{O}\left(N(\ln N)^{d-2} dwp \ln p \right)$ and depth of $\mathcal{O}\left(l \ln p + N^2 \right)$, or have $\mathcal{O}\left(dwp \ln p \right)$ width and $\mathcal{O}\left(N(\ln N)^{d-2} l \ln p + N^3 (\ln N)^{d-2} \right)$ depth. It is obvious that the non-zero parameters for $M_{\mathcal{F}_N}$ is in the order of

$$\mathcal{O}\left(N (\ln N)^{d-2} \left(dw^2 p \ln p + \|\mathbf{n}\|_2^2 \right) \right) \leq \mathcal{O}\left(N (\ln N)^{d-2} \left(dw^2 p \ln p + N^2 \right) \right).$$

We now finish the proof. □

A.4. Proof of Theorem 3

Proof. As shown in Eq. (2) that $\mathbf{x}_{k+1} = \left[\mathcal{L}_{k+1}(\mathbf{x}_k);\ \tilde{\mathcal{A}}_{k+1}(\mathbf{x}_0) + \max\left\{\mathcal{W}_{k+1}(\mathbf{x}_k), \sigma_{k+1}\left(\mathcal{A}_{k+1}\left(\mathbf{x}_{\hat{k}}\right)\right)\right\}\right]$, by introducing an auxiliary variable $\mathbf{y}_k$, MCN can be reformulated as a nested function as follows:

$$\mathbf{y}_{k+1} = \mathcal{G}_k(\mathbf{y}_k) = \operatorname{concate}\left(\mathbf{y}_k, \sigma_{\mathrm{MCN}}\left(\mathcal{T}_{k+1}(\mathbf{y}_k)\right)\right) = \left[\mathbf{y}_k; \sigma_{\mathrm{MCN}}\left(\mathcal{T}_{k+1}(\mathbf{y}_k)\right)\right], \qquad \mathbf{y}_0 = \mathbf{x}_0, \tag{7}$$

where $\mathcal{T}_{k+1}(\cdot)$ is a Block Sparse Operator Matrix and $\mathbf{y}_k$ is a column vector consist of all entries from $\mathbf{x}_0$ to $\mathbf{x}_k$, which are defined as follows:

$$\mathcal{T}_{k+1}(\cdot) = \begin{bmatrix} \tilde{\mathcal{A}}_{k+1} & \dots & \mathcal{O} & \dots & \mathcal{O} \\ \mathcal{O} & \dots & \mathcal{A}_{k+1} & \dots & \mathcal{O} \\ \mathcal{O} & \dots & \mathcal{O} & \dots & \mathcal{W}_{k+1} \\ \mathcal{O} & \dots & \mathcal{O} & \dots & \mathcal{L}_{k+1} \end{bmatrix}, \qquad \mathbf{y}_k = \begin{bmatrix} \mathbf{x}_0 \\ \vdots \\ \mathbf{x}_k \end{bmatrix}.$$

It should be mentioned that, each row of $\mathcal{T}_{k+1}(\cdot)$ only has one non-zero block at $\hat{k}$-th column, the index of which is determined by the structure of each MCN block. And we use a concatenate vector $\mathbf{y}_k$ to integrate different subscripts $\hat{k}$.

Moreover, σ_{MCN} in Eq. (7) is a special activation function corresponding to Eq. (2):

$$\sigma_{\mathrm{MCN}}\left(\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}\right) = \left[d; a + \max\left\{\sigma_{k+1}(b), c\right\}\right].$$

Claim 9. *The operators $\mathcal{G}_k(\cdot)$, σ_{MCN}, and $\mathcal{T}_{k+1}$ in Eq. (7) are Lipschitz continuous w.r.t. ℓ_1 norm. Moreover, the Lipschitz constant for the operator $\mathcal{G}_k(\cdot)$ is*

$$\kappa_k \coloneqq \left(1 + \max\{\rho_{k+1}, 2\} \max\{\|\tilde{\mathcal{A}}_{k+1}\|_1, \|\mathcal{A}_{k+1}\|_1, \|\mathcal{W}_{k+1} + \mathcal{L}_{k+1}\|_1\}\right),$$

where $\|\cdot\|_1$ is the operator ℓ_1 norms induced by vector ℓ_1 norms

$$\|\mathcal{A}\|_1 = \max_{x \neq 0} \frac{\|\mathcal{A}(\mathbf{x})\|_1}{\|\mathbf{x}\|_1}.$$

Proof. Let $\mathbf{y}_k = \begin{bmatrix} \mathbf{x}_0 \\ \vdots \\ \mathbf{x}_k \end{bmatrix}$ and $\mathbf{y}'_k = \begin{bmatrix} \mathbf{x}'_0 \\ \vdots \\ \mathbf{x}'_k \end{bmatrix}$, then we have $\mathcal{T}_{k+1}(\mathbf{y}_k) = \begin{bmatrix} \tilde{\mathcal{A}}_{k+1}(\mathbf{x}_0) \\ \mathcal{A}_{k+1}(\mathbf{x}_{\hat{k}}) \\ \mathcal{W}_{k+1}(\mathbf{x}_k) \\ \mathcal{L}_{k+1}(\mathbf{x}_k) \end{bmatrix}$ and $\mathcal{T}_{k+1}(\mathbf{y}_k) = \begin{bmatrix} \tilde{\mathcal{A}}_{k+1}(\mathbf{x}'_0) \\ \mathcal{A}_{k+1}(\mathbf{x}'_{\hat{k}}) \\ \mathcal{W}_{k+1}(\mathbf{x}'_k) \\ \mathcal{L}_{k+1}(\mathbf{x}'_k) \end{bmatrix}$.

It can be seen that $\mathcal{T}_{k+1}(\cdot)$ is a Lipschitz continuous function w.r.t. ℓ_1 norm. By the definition of Lipschitz continuity and induction norm, it is easy to check that $L_{\mathcal{T}_{k+1}} = \max\{\|\tilde{\mathcal{A}}_{k+1}\|_1, \|\mathcal{A}_{k+1}\|_1, \|\mathcal{W}_{k+1} + \mathcal{L}_{k+1}\|_1\}$.

For convenience, we use $\mathbf{p}, \mathbf{q}, \mathbf{r}, \mathbf{s}$ and $\mathbf{p}', \mathbf{q}', \mathbf{r}', \mathbf{s}'$ to denote each entries of $\mathcal{T}_{k+1}(\mathbf{y}_k)$ and $\mathcal{T}_{k+1}(\mathbf{y}'_k)$, which means $\mathcal{T}_{k+1}(\mathbf{y}_k) = \begin{bmatrix} \mathbf{p} \\ \mathbf{q} \\ \mathbf{r} \\ \mathbf{s} \end{bmatrix}$ and $\mathcal{T}_{k+1}(\mathbf{y}'_k) = \begin{bmatrix} \mathbf{p}' \\ \mathbf{q}' \\ \mathbf{r}' \\ \mathbf{s}' \end{bmatrix}$. Then by using the definition of Lipschitz continuous, we have

$$\begin{aligned}
\mathcal{G}_k(\mathbf{y}_k) - \mathcal{G}_k(\mathbf{y}'_k) &= \left[\mathbf{y}_k; \sigma_{\mathrm{MCN}}\left(\mathcal{T}_{k+1}(\mathbf{y}_k)\right)\right] - \left[\mathbf{y}'_k; \sigma_{\mathrm{MCN}}\left(\mathcal{T}_{k+1}(\mathbf{y}'_k)\right)\right] \\
&= \left[\mathbf{y}_k - \mathbf{y}'_k; \sigma_{\mathrm{MCN}}\left(\mathcal{T}_{k+1}(\mathbf{y}_k)\right) - \sigma_{\mathrm{MCN}}\left(\mathcal{T}_{k+1}(\mathbf{y}'_k)\right)\right] \\
&= \left[\mathbf{y}_k - \mathbf{y}'_k; \mathbf{s} - \mathbf{s}'; (\mathbf{p} - \mathbf{p}') + \left(\max\left\{\sigma_{k+1}(\mathbf{q}), \mathbf{r}\right\} - \max\left\{\sigma_{k+1}(\mathbf{q}'), \mathbf{r}'\right\}\right)\right] \\
&= \left[\mathbf{y}_k - \mathbf{y}'_k; \mathbf{s} - \mathbf{s}'; (\mathbf{p} - \mathbf{p}') + \left(\max\left\{\sigma_{k+1}(\mathbf{q}), \mathbf{r}\right\} - \max\left\{\sigma_{k+1}(\mathbf{q}'), \mathbf{r}'\right\}\right)\right] \\
&= \left[\mathbf{y}_k - \mathbf{y}'_k; \mathbf{s} - \mathbf{s}'; (\mathbf{p} - \mathbf{p}') + \left(\mathrm{ReLU}\left\{\sigma_{k+1}(\mathbf{q}) - \mathbf{r}\right\} + \mathbf{r} - \mathrm{ReLU}\left\{\sigma_{k+1}(\mathbf{q}') - \mathbf{r}'\right\} - \mathbf{r}'\right)\right] \\
&= \left[\mathbf{y}_k - \mathbf{y}'_k; \mathbf{s} - \mathbf{s}'; (\mathbf{p} - \mathbf{p}') + (\mathbf{r} - \mathbf{r}') + \left(\mathrm{ReLU}\left\{\sigma_{k+1}(\mathbf{q}) - \mathbf{r}\right\} - \mathrm{ReLU}\left\{\sigma_{k+1}(\mathbf{q}') - \mathbf{r}'\right\}\right)\right]
\end{aligned}$$

Then

$$
\begin{aligned}
&\|\mathcal{G}_k(\mathbf{y}_k) - \mathcal{G}_k(\mathbf{y}'_k)\|_1 \\
=&\left|[\mathbf{y}_k - \mathbf{y}'_k; \mathbf{s} - \mathbf{s}'; (\mathbf{p} - \mathbf{p}') + (\mathbf{r} - \mathbf{r}') + (\mathrm{ReLU}\left\{\sigma_{k+1}(\mathbf{q}) - \mathbf{r}\right\} - \mathrm{ReLU}\left\{\sigma_{k+1}(\mathbf{q}') - \mathbf{r}'\right\})]\right|_1 \\
=&\|\mathbf{y}_k - \mathbf{y}'_k\|_1 + \|\mathbf{s} - \mathbf{s}'|_1 + |(\mathbf{p} - \mathbf{p}') + (\mathbf{r} - \mathbf{r}') + (\mathrm{ReLU}\left\{\sigma_{k+1}(\mathbf{q}) - \mathbf{r}\right\} - \mathrm{ReLU}\left\{\sigma_{k+1}(\mathbf{q}') - \mathbf{r}'\right\})\|_1 \\
\leq&\|\mathbf{y}_k - \mathbf{y}'_k\|_1 + \|\mathbf{s} - \mathbf{s}'\|_1 + \|\mathbf{p} - \mathbf{p}'\|_1 + \|\mathbf{r} - \mathbf{r}'\|_1 + \|\mathrm{ReLU}\left\{\sigma_{k+1}(\mathbf{q}) - \mathbf{r}\right\} - \mathrm{ReLU}\left\{\sigma_{k+1}(\mathbf{q}') - \mathbf{r}'\right\}\|_1 \\
\leq&\|\mathbf{y}_k - \mathbf{y}'_k\|_1 + \|\mathbf{s} - \mathbf{s}'\|_1 + \|\mathbf{p} - \mathbf{p}'\|_1 + \|\mathbf{r} - \mathbf{r}'\|_1 + \|(\sigma_{k+1}(\mathbf{q}) - \mathbf{r}) - (\sigma_{k+1}(\mathbf{q}') - \mathbf{r}')\|_1 \\
\leq&\|\mathbf{y}_k - \mathbf{y}'_k\|_1 + \|\mathbf{s} - \mathbf{s}'\|_1 + \|\mathbf{p} - \mathbf{p}'\|_1 + 2\|\mathbf{r} - \mathbf{r}'\|_1 + \|\sigma_{k+1}(\mathbf{q}) - \sigma_{k+1}(\mathbf{q}')\|_1
\end{aligned}
$$

Suppose that the activation function σ_{k+1} is also Lipschitz continuous with a Lipschitz constant ρ_{k+1}, then we have

$$
\begin{aligned}
&\|\mathcal{G}_k(\mathbf{y}_k) - \mathcal{G}_k(\mathbf{y}'_k)\|_1 \\
\leq&\|\mathbf{y}_k - \mathbf{y}'_k\|_1 + \|\mathbf{s} - \mathbf{s}'\|_1 + \|\mathbf{p} - \mathbf{p}'\|_1 + 2\|\mathbf{r} - \mathbf{r}'\|_1 + \|\sigma_{k+1}(\mathbf{q}) - \sigma_{k+1}(\mathbf{q}')\|_1 \\
\leq&\|\mathbf{y}_k - \mathbf{y}'_k\|_1 + \|\mathbf{s} - \mathbf{s}'\|_1 + \|\mathbf{p} - \mathbf{p}'\|_1 + 2\|\mathbf{r} - \mathbf{r}'\|_1 + \rho_{k+1}\|\mathbf{q} - \mathbf{q}'\|_1 \\
\leq&\|\mathbf{y}_k - \mathbf{y}'_k\|_1 + \max\{\rho_{k+1}, 2\}\|\mathcal{T}_{k+1}(\mathbf{y}_k) - \mathcal{T}_{k+1}(\mathbf{y}'_k)\|_1 \\
\leq&\|\mathbf{y}_k - \mathbf{y}'_k\|_1 + \max\{\rho_{k+1}, 2\}L_{\mathcal{T}_{k+1}}\|\mathbf{y}_k - \mathbf{y}'_k\|_1 \\
=&\left(1 + \max\{\rho_{k+1}, 2\}L_{\mathcal{T}_{k+1}}\right)\|\mathbf{y}_k - \mathbf{y}'_k\|_1 \\
=&\left(1 + \max\{\rho_{k+1}, 2\}\max\{\|\tilde{\mathcal{A}}_{k+1}\|_1, \|\mathcal{A}_{k+1}\|_1, \|\mathcal{W}_{k+1} + \mathcal{L}_{k+1}\|_1\}\right)\|\mathbf{y}_k - \mathbf{y}'_k\|_1
\end{aligned}
\tag{8}
$$

From the above, it is easy to get that $\mathcal{G}_k(\cdot)$, σ_{MCN}, and $\mathcal{T}_{k+1}$ in Eq. (7) are all Lipschitz functions w.r.t. ℓ_1 norm. □

Now, given the parameters $\boldsymbol{\theta}$ of MCN, we define

$$
\mathcal{G}_{i\to j}(\boldsymbol{\theta}) \coloneqq \mathcal{G}_j \circ \cdots \mathcal{G}_i \circ, \quad 1 \leq i \leq j \leq l,
$$

and

$$
\sigma_{\text{C-MCN}}(\mathbf{x}) = [\mathbf{x}; \sigma_{\text{MCN}}(\mathbf{x})].
$$

For given $\varepsilon > 0$, we consider two MCNs $\mathbf{f}_{\boldsymbol{\theta}_1}$ and $\mathbf{f}_{\boldsymbol{\theta}_2}$ that both are from $\mathcal{F}(\boldsymbol{\theta}, S)$ such that $\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_1 \leq \varepsilon$,

$$
\begin{aligned}
&\|\mathbf{f}_{\boldsymbol{\theta}_1}(\mathbf{x}) - \mathbf{f}_{\boldsymbol{\theta}_2}(\mathbf{x})\|_1 \overset{(a)}{=} \left\|\sum_{k=1}^{l}\mathcal{G}_{k+1\to L}(\boldsymbol{\theta}_1) \circ \sigma_{\text{C-MCN}}\left(\left(\mathcal{T}_{k+1}^{(\boldsymbol{\theta}_1)} - \mathcal{T}_{k+1}^{(\boldsymbol{\theta}_2)}\right)(\mathcal{G}_{1\to k}(\boldsymbol{\theta}_2) \circ \mathbf{x})\right)\right\|_1 \\
\leq&\sum_{k=1}^{l}\prod_{i=k+1}^{l}\kappa_i\rho\left\|\left(\mathcal{T}_{k+1}^{(\boldsymbol{\theta}_1)} - \mathcal{T}_{k+1}^{(\boldsymbol{\theta}_2)}\right)(\mathcal{G}_{1\to k}(\boldsymbol{\theta}_2) \circ \mathbf{x})\right\|_1 \leq \rho\sum_{k=1}^{l}\prod_{i=k+1}^{l}\kappa_i\|(\mathcal{G}_{1\to k}(\boldsymbol{\theta}_2) \circ \mathbf{x})\|_1\varepsilon \\
\leq&\rho\varepsilon\sum_{k=1}^{l}\prod_{i=k+1}^{l}\kappa_i\|(\mathcal{G}_{1\to k}(\boldsymbol{\theta}_2) \circ \mathbf{x})\|_1 \leq \rho\varepsilon\sum_{k=1}^{l}\prod_{i=1}^{l}\kappa_i\|\mathbf{x}\|_1 \leq \rho l\|\mathbf{x}\|_1\prod_{i=1}^{l}\kappa_i\varepsilon.
\end{aligned}
$$

where (a) comes from the Telescoping sum. Thus, for a fixed sparsity pattern S (i.e., the location of nonzero elements in $\boldsymbol{\theta}$), the covering number is bounded by

$$
\left(\frac{\rho l\|\mathbf{x}\|_1\prod_{i=1}^{l}l_i}{\delta}\right)^s.
$$

Since the number of the sparsity patterns is bounded by $\binom{w^2l}{s} \leq (w+1)^{ls}$, the log of covering number is bounded above by

$$
\ln\left((w+1)^{ls}\left(\frac{\rho l\|\mathbf{x}\|_1\prod_{i=1}^{l}\kappa_i}{\delta}\right)^s\right) \leq \mathcal{O}\left(ls\ln\left(\frac{\rho\|\mathbf{x}\|_1\prod_{i=1}^{l}\kappa_i}{\delta}\right)\right).
$$

□

A.5. Proof of Theorem 4

Proof. Without loss of generality, we assume $d_y = 1$ and let the smoothness parameter $\beta = 1$ in this proof, and the proof can be easily extended to high dimensional and general β case . We denote the estimator $\mathbf{f}_{\boldsymbol{\theta}}$ as $f_{\boldsymbol{\theta}}$ in the following. We denote by g the target function, since it is smooth, we assume that g has bounded derivative. We also let the compact set $\mathcal{C}$ be the input domain in this proof.

Since the objective $L(\boldsymbol{\theta})$ obtains its optimum value on the training set, MCN fits all the training data, i.e., $f_{\boldsymbol{\theta}}(\mathbf{x}_i) = y_i,\ \forall i \in [n]$. Hence, $f_{\boldsymbol{\theta}}$ is an estimator that interpolates the training data.

In the exactly fitting case, we know that $f_{\boldsymbol{\theta}}$ partitions the compact set $\mathcal{C}$ into many nondegenerate subsets. On each subset $\mathcal{C}_s$, we have

$$f_{\boldsymbol{\theta}}(\mathbf{x}) = \mathbf{w}_s^\top \sigma\left(\mathbf{A}_s \mathbf{x}\right) + \mathbf{b}_s^\top \mathbf{x}, \quad \forall \mathbf{x} \in \mathcal{C}_s,$$

where $\mathbf{A}_s$ has different shapes for different subsets $\mathcal{C}_s$, for brevity, we let $\mathbf{A}_s \in \mathbb{R}^{d_a \times d_x}$. Each $\mathbf{x} \in \mathcal{C}$ is contained in at least one of these subsets; let $\mathcal{V}(\mathbf{x})$ denote the set of training data points $\{\mathbf{x}_{(1)}, \cdots, \mathbf{x}_{(|v|)}\}$ that determine the function surface of $f_{\boldsymbol{\theta}}(\cdot)$ on this subset $\mathcal{C}_s$ containing $\mathbf{x}$, where $d_x \leq |v| \leq (d_x+1)d_a + d_x$.

Consider the following linear equation:

$$\begin{bmatrix} g(\mathbf{x}_{(1)}) + \varepsilon_1 & \cdots & g(\mathbf{x}_{(|v|)}) + \varepsilon_{|v|} \\ 1 & \cdots & 1 \end{bmatrix} \overline{\mathbf{w}} = \begin{bmatrix} f_{\boldsymbol{\theta}}(\mathbf{x}) \\ 1 \end{bmatrix}.$$

where ε_i's are the noise terms and are i.i.d. Gassuian.

Claim 10. *With high probability, $\overline{\mathbf{w}}$ exists and for some constant $C_w > 0$, we have:*

$$\|\overline{\mathbf{w}}\|^2 \leq \frac{C_w}{|v|}. \tag{9}$$

Proof. Let

$$\begin{bmatrix} g(\mathbf{x}_{(1)}) + \varepsilon_1 & \cdots & g(\mathbf{x}_{(|v|)}) + \varepsilon_{|v|} \\ 1 & \cdots & 1 \end{bmatrix} := \begin{bmatrix} \mathbf{G} \\ \mathbf{1} \end{bmatrix},$$

and denote $\sigma_{\min}(\mathbf{G})$ as the minimal singular value of the matrix $\mathbf{G}$. By Corollary 3.1.3 of (Horn & Johnson, 1991), we have:

$$\sigma_{\min}\left(\begin{bmatrix} \mathbf{G} \\ \mathbf{1} \end{bmatrix}\right) \geq \sigma_{\min}\left(\mathbf{G}\right).$$

Note that the column of matrix $\mathbf{G}$ is bounded by C_G (i.e., the ℓ_2-norm of each column is upper bounded), without loss of generality, we assume that C_G is small, otherwise, we can divide all the function values $g(\mathbf{x})$ by a large constant. Moreover, w.o.l.g. we let $\mathbb{E}[\mathbf{g}(\cdot)] = \mathbf{I}_{d_y}$. i.e., each dimension of $\mathbf{g}(\cdot)$ is independent. We also note that the columns of matrix $\mathbf{G}$ are also independent with each other, then according to Theorem 5.41 in (Vershynin, 2010), with probability at least $1 - 2\exp\left(-ct^2\right)$, we have:

$$\sigma_{\min}\left(\mathbf{G}\right) \geq \sqrt{|v|} - tC_G > 0,$$

where the last inequality holds when C_G is small and $|v|$ is large, for convenience, we let $\left(\sqrt{|v|} - tC_G\right) \geq \sqrt{|v|}/2$. We can conclude that with high probability:

$$\sigma_{\min}\left(\begin{bmatrix} \mathbf{G} \\ \mathbf{1} \end{bmatrix}\right) \geq \sqrt{|v|} - tC_G \geq \frac{\sqrt{|v|}}{2} > 0.$$

Namely, $\overline{\mathbf{w}}$ exists and

$$\overline{\mathbf{w}} = \begin{bmatrix} \mathbf{G} \\ \mathbf{1} \end{bmatrix}^\dagger \begin{bmatrix} f_{\boldsymbol{\theta}}(\mathbf{x}) \\ 1 \end{bmatrix},$$

where $\mathbf{A}^\dagger$ represents the pseudo-inverse of the matrix $\mathbf{A}$. Then we have:

$$\|\overline{\mathbf{w}}\|^2 \leq \left\| \begin{bmatrix} \mathbf{G} \\ \mathbf{1} \end{bmatrix}^\dagger \right\|^2 \left\| \begin{bmatrix} f_{\boldsymbol{\theta}}(\mathbf{x}) \\ 1 \end{bmatrix} \right\|^2 \leq \frac{4C_f}{|v|},$$

where $C_f = 1 + \max_{\mathbf{x}\in\mathcal{C}} \|f_{\boldsymbol{\theta}}(\mathbf{x})\|$, and $\|\cdot\|$ is the spectral norm for matrix and ℓ_2-norm for vector. We finish the proof of this claim. □

By $\overline{\mathbf{w}}$, we can represent $f_{\boldsymbol{\theta}}(\mathbf{x})$ as a linear combination way:

$$f_{\boldsymbol{\theta}}(\mathbf{x}) = \sum_{i=1}^{|v|} \overline{w}_i \, g(\mathbf{x}_{(i)}) := \sum_{i=1}^{n} \mathbf{I}\left\{\mathbf{x}_i \in \mathcal{V}(\mathbf{x})\right\} W(\mathbf{x}, \mathbf{x}_i)(g(\mathbf{x}_i) + \varepsilon_i),$$

where $W(\mathbf{x}, \mathbf{x}_i) := \overline{w}_i$, $W : \mathbb{R}_x^d \times \mathbb{R}_x^d \to \mathbb{R}$ is a coefficient mapping and $g(\cdot)$ is the target function. Note that, for any $\mathbf{x}$, $\sum_{i=1}^{|v|} \overline{w}_i = 1$ indicates:

$$\sum_{i=1}^{n} \mathbf{I}\left\{\mathbf{x}_i \in \mathcal{V}(\mathbf{x})\right\} W(\mathbf{x}, \mathbf{x}_i) = 1. \tag{10}$$

Hence, for all $\mathbf{x} \in \mathcal{C}$, we can have:

$$\sum_{i=1}^{n} \left(\mathbf{I}\left\{\mathbf{x}_i \in \mathcal{V}(\mathbf{x})\right\} W(\mathbf{x}, \mathbf{x}_i) g(\mathbf{x})\right) = g(\mathbf{x}). \tag{11}$$

We consider the event:

$$\mathcal{E} := \left\{\operatorname{diam}(\mathcal{C}_s) \le h\right\},$$

where we specify the scale of h at the last of this proof. Since the points $\{\mathbf{x}_i\}_{i=1}^n \setminus \mathcal{V}(\mathbf{x})$ are out of the subset $\mathcal{C}_s$, we can observe that:

$$p(\overline{\mathcal{E}}) \le \left(1 - C_1 p_{\min} h^{d_x}\right)^{n-|v|} \le \exp\left\{-C_2 p_{\min} n h^{d_x}\right\},$$

where the last inequality comes from $|v| \ll n$ and C_1 and $C_2 > 0$ is a constant which is independent of size n. On the event $\overline{\mathcal{E}}$, due to the bounded first derivative of $g(\cdot)$ and Eq. (11), with probability at least $1 - 2\exp\left(-c_\varepsilon^2/2\right)$, we have:

$$|f_{\boldsymbol{\theta}}(\mathbf{x}) - g(\mathbf{x})| = \left|\sum_{i=1}^{n} \left(\mathbf{I}\left\{\mathbf{x}_i \in \mathcal{V}(\mathbf{x})\right\} W(\mathbf{x}, \mathbf{x}_i)(g(\mathbf{x}_i) + \varepsilon_i - g(\mathbf{x}))\right)\right| \le C_{g'} \operatorname{diam}(\mathcal{C}) + c_\varepsilon := C_g.$$

Thus, the contribution of event $\overline{\mathcal{E}}$ to generalization bound is at most $C_g^2 \exp\left\{-C p_{\min} h^{d_x}\right\}$, a lower-order term compared to the remaining contribution of event $\mathcal{E}$.

By the event $\mathcal{E}$, we have the following decomposition:

$$\mathbb{E}\left[\left|f_{\boldsymbol{\theta}}(\mathbf{x}) - g(\mathbf{x})\right|^2\right] \le \underbrace{\mathbb{E}\left[\left|f_{\boldsymbol{\theta}}(\mathbf{x}) - g(\mathbf{x})\right|^2 \mathbf{I}\left\{\mathcal{E}\right\}\right]}_{B^2(\mathbf{x})} + C_g^2 \exp\left\{-C p_{\min} h^{d_x}\right\},$$

where $\mathbb{E}[\cdot] := \mathbb{E}_{\mathcal{S}^n}\left[\mathbb{E}_\varepsilon[\cdot \mid \mathcal{S}^n]\right]$.

In the following, we provide the generalization bound of the bias term $B^2(\mathbf{x})$. Due to Eq. (10), we have:

$$B^2(\mathbf{x}) = \mathbb{E}\left[\sum_{i,j=1}^{n} \left(g(\mathbf{x}_i) + \varepsilon_i - g(\mathbf{x})\right)\left(g(\mathbf{x}_j) + \varepsilon_j - g(\mathbf{x})\right) W_i W_j \mathbf{I}\left\{\mathcal{E}\right\}\right],$$

where

$$W_i = \mathbf{I}\left\{\mathbf{x}_i \in \mathcal{V}(\mathbf{x})\right\} W(\mathbf{x}, \mathbf{x}_i).$$

Due to the event $\{\mathbf{x}_i \in \mathcal{V}(\mathbf{x})\}$, we can conclude that $\|\mathbf{x}_i - \mathbf{x}\| \le h$ and by the bounded first derivative of $g(\cdot)$, we get:

$$B^2(\mathbf{x}) \le C_{g'}^2 h^2 \sum_{i,j=1}^{n} \mathbb{E}\left[W_i W_j \mathbf{I}\left\{\mathcal{E}\right\}\right] + \sum_{i,j=1}^{n} \mathbb{E}\left[\varepsilon_i \varepsilon W_i W_j \mathbf{I}\left\{\mathcal{E}\right\}\right].$$

Note that

$$\sum_{i,j=1}^{n} \mathbb{E}\left[\varepsilon_i \varepsilon W_i W_j \mathbf{I}\left\{\mathcal{E}\right\}\right] = \sum_{i=1}^{n} \mathbb{E}\left[W_i^2 \mathbf{I}\left\{\mathcal{E}\right\}\right] := \sigma_n.$$

In general, according to correlation between W_i and W_j, we decompose the sum term:

$$\sum_{i,j=1}^{n}\mathbb{E}\left[W_iW_j\mathbf{I}\left\{\mathcal{E}\right\}\right]=\underbrace{\sum_{i=1}^{n}\mathbb{E}\left[W_i^2\mathbf{I}\left\{\mathcal{E}\right\}\right]}_{\sigma_n}+\sum_{i\neq j}^{n}\mathbb{E}\left[W_iW_j\mathbf{I}\left\{\mathcal{E}\right\}\right]$$

$$=\quad\mathbb{E}\left[\sum_{i\neq j}^{n}W_iW_j\mathbf{I}\left\{\mathcal{E}\right\}\right]+\sigma_n\leq\mathbb{E}\left[(\sum_{i}^{n}W_i)^2\mathbf{I}\left\{\mathcal{E}\right\}\right]+\sigma_n\leq 1+\sigma_n,$$

where the last inequality comes from Eq. (10). On one hand, we have:

$$\sigma_n=\sum_{i=1}^{|v|}\mathbb{E}\left[W(\mathbf{x},\mathbf{x}_{(i)})^2\mathbf{I}\left\{\mathcal{E}\right\}\right]=\mathbb{E}\left[\|\overline{\mathbf{w}}\|^2\mathbf{I}\left\{\mathcal{E}\right\}\right].$$

Actually, the random variables $\mathbf{I}\left\{\mathbf{x}_i\in\mathcal{V}(\mathbf{x})\right\}$ follow the Bernoulli distribution with parameter:

$$\hat{p}:=P(\mathbf{x}_i\in\mathcal{V}(x))\geq c_0p_{\min}h^{d_x},$$

where $c_0>0$ depends on the shape of set $\mathcal{C}_s$ and d. Hence we can divide the exception into two term:

$$\mathbb{E}\left[\|\overline{\mathbf{w}}\|^2\mathbf{I}\left\{\mathcal{E}\right\}\right]\leq\quad\underbrace{\mathbb{E}\left[\|\overline{\mathbf{w}}\|^2\mathbf{I}\left\{\mathcal{E}\right\}\mathbf{I}\left\{|v|<\frac{n\hat{p}}{2}\right\}\right]}_{E_1}+\underbrace{\mathbb{E}\left[\|\overline{\mathbf{w}}\|^2\mathbf{I}\left\{\mathcal{E}\right\}\mathbf{I}\left\{|v|\geq\frac{n\hat{p}}{2}\right\}\right]}_{E_2}.$$

For E_2, together with Eq. (9), we have:

$$\begin{aligned}E_2\leq&\quad p_{\max}\frac{C_w}{|v|}\int_{\mathcal{C}_s}\mathbf{I}\left\{\mathcal{E}\right\}d\mathbf{x}\leq\quad cp_{\max}\frac{2C_w}{n\hat{p}}h^{d_x}\int_0^1 r^{d_x-1}dr\\&\leq\quad cp_{\max}\frac{2C_w}{c_0np_{\min}}:=\frac{c_1}{n},\end{aligned}$$

where $c>0$ is the constant which is independent of n and depends on the shape of the set $\mathcal{C}_s$. For E_1, we have

$$\begin{aligned}E_1\leq&\quad p_{\max}\frac{C_w}{|v|}\mathbb{E}\left[\mathbf{I}\left\{|v|<\frac{n\hat{p}}{2}\right\}\right]\leq\quad p_{\max}\frac{C_w}{d_x}P\left(\sum_{i=1}^{n}\mathbf{I}\left\{\mathbf{x}_i\in\mathcal{V}(\mathbf{x})\right\}<\frac{n\hat{p}}{2}\right)\\=&\quad p_{\max}\frac{C_w}{d_x}P\left(\left|\sum_{i=1}^{n}\mathbf{I}\left\{\mathbf{x}_i\in\mathcal{V}(\mathbf{x})\right\}-n\hat{p}\right|>\frac{n\hat{p}}{2}\right)\\\overset{(a)}{\leq}&\quad p_{\max}\frac{C_w}{d_x}\exp\left\{\frac{(n\hat{p}/2)^2}{2n\hat{p}(1-\hat{p})+n\hat{p}/3}\right\}\leq\exp\left\{-c_2nh^{d_x}\right\},\end{aligned}$$

where (a) comes from the Bernstein's inequality.

Combing all the above results together, by setting $h=\mathcal{O}\left(n^{-\frac{1}{d_x+2\beta}}\right)$, we obtain:

$$\begin{aligned}\mathbb{E}\left[|f_{\boldsymbol{\theta}}(\mathbf{x})-g(\mathbf{x})|^2\right]\leq&\quad C_g^2\exp\left\{-C_2p_{\min}nh^{d_x}\right\}+C_{g'}^2h^2\cdot\left(1+\frac{c_1}{n}+\exp\left\{-c_2nh^{d_x}\right\}\right)+\left(\frac{c_1}{n}+\exp\left\{-c_2nh^{d_x}\right\}\right)\\\leq&\quad C_3\exp\left\{-C_4nh^{d_x}\right\}+C_5h^2+\frac{c_1}{n}\\\leq&\quad\frac{C_3C_4}{nh^{d_x}}+C_5h^2+\frac{c_1}{n}\leq\ C_6n^{(-\frac{2}{2+d_x})},\end{aligned}$$

where $\{C_3,C_4,C_5,C_6\}>0$ are universal constants and the last inequality holds when $h=\mathcal{O}\left(n^{-\frac{1}{d_x+2\beta}}\right)$. It is obvious that, when n is large enough and the data is sampled uniformly, the event $\mathcal{E}:=\{\mathrm{diam}(\mathcal{C}_s)\leq h\}$ can easily happen for $h=\mathcal{O}\left(n^{-\frac{1}{d_x+2\beta}}\right)$.

We now finish the proof of this theorem. □

A.6. Proof of Theorem 5

Proof. Actually, the proof is very direct. Let $\ell_\Phi(\cdot) := \ell(\Phi(\cdot), \mathbf{y})$ and $\nabla\ell_\Phi(\mathbf{h}(\mathbf{x}))$ be the gradient $\nabla\ell_\Phi$ evaluated at $\mathbf{h}(\mathbf{x})$. Denote by $\boldsymbol{\theta}_0$ the parameters of $\mathbf{h}_0$. Note that $\mathbf{h}(\cdot)$ the DNN appended with l-layer MCN has the parameters $[\boldsymbol{\theta}_0, \boldsymbol{\theta}_l]$. Given the local minimum $\left[\tilde{\boldsymbol{\theta}}_0, \tilde{\boldsymbol{\theta}}_{l+1}\right]$ and the parameters $\boldsymbol{\theta}_0'$ such that $\mathbf{h}_0(\cdot \mid \boldsymbol{\theta}_0')$ is injective w.r.t to the input $\mathbf{x}$, then we have

$$\frac{1}{n}\sum_{i=1}^{n}\ell_\Phi\left(\mathbf{h}\left(\mathbf{x}_i \mid \left[\tilde{\boldsymbol{\theta}}_0, \tilde{\boldsymbol{\theta}}_{l+1}\right]\right)\right) \le \min_{[\boldsymbol{\theta}_0, \boldsymbol{\theta}_l]}\frac{1}{n}\sum_{i=1}^{n}\ell_\Phi\left(\mathbf{h}(\mathbf{x}_i)\right) \le \min_{\boldsymbol{\theta}_l}\frac{1}{n}\sum_{i=1}^{n}\ell_\Phi\left(\mathbf{h}\left(\mathbf{x}_i \mid \boldsymbol{\theta}_0'\right)\right),$$

where the first inequity comes from Theorem 1. It is obvious the right side in the above inequality is the loss of a l-layer MCN with the set $\{(\mathbf{h}(\mathbf{x}_i \mid \boldsymbol{\theta}_0'), y_i)\}_{i=1}^n$ at the global minimum. The problem becomes a learning target with the input as $\mathbf{h}(\mathbf{x}_i \mid \boldsymbol{\theta}_0')$. As shown in Claim 3, a $(n-1)$-th order polynomial can exactly fit the training set. For a given polynomial, it is easy to modified it to make it satisfy the Condition 1, e.g., extending and rescaling. With the virtue of Theorem 2, MCN can approximate the functions satisfing Condition 1 arbitrarily well as it goes deeper and wider. Hence, we have

$$\min_{\boldsymbol{\theta}_l}\frac{1}{n}\sum_{i=1}^{n}\ell_\Phi\left(\mathbf{h}(\mathbf{x}_i \mid \boldsymbol{\theta}_0')\right) \to 0, \quad l \to \infty.$$

Thus,

$$\frac{1}{n}\sum_{i=1}^{n}\ell_\Phi\left(\mathbf{h}\left(\mathbf{x}_i \mid \left[\tilde{\boldsymbol{\theta}}_0, \tilde{\boldsymbol{\theta}}_{l+1}\right]\right)\right) \to 0$$

holds at any local minimum $\left[\tilde{\boldsymbol{\theta}}_0, \tilde{\boldsymbol{\theta}}_{l+1}\right]$ as MCN goes deeper and wider. □

B. Connection to Linear Regression

In this section, we shall quantitatively describe the quality of each local minimum on the regression task. Denote $\mathcal{P}_{\mathbf{D}}$ as the orthogonal projection matrix onto the column space (or range space) of a matrix $\mathbf{D}$, thereby $\mathcal{P}_{\mathbf{D}}^{\perp} = \mathbf{I} - \mathcal{P}_{\mathbf{D}}$. Let $\otimes$ represent the Kronecker product, let $\mathrm{vec}(\cdot)$ be the vectorization of a matrix, and denote the d_y-dimension identify matrix as $\mathbf{I}_{d_y}$. Denote by $\mathbf{Y} := [\mathbf{y}_1, \cdots, \mathbf{y}_n]$ the target matrix. With these notations, we have the following theorem to measure the training objective quantitatively.

Theorem 9 (Monotonicity of Objective)**.** *Suppose that $\boldsymbol{\theta}_l$ is a local minimum to problem (3), in which the loss ℓ is chosen as the squared loss and the mapping $\Psi(\cdot)$ is a learnable matrix of size $d_y \times d_l$. Then the following holds:*

(i) There exists a matrix $\mathbf{D}$ whose column space expands, as the depth and width of MCN increase; and

$$L(\boldsymbol{\theta}_l) = \frac{1}{n}\|\mathcal{P}_{\mathbf{D}}^{\perp}\,\mathrm{vec}\,(\mathbf{Y})\,\|^2.$$

(ii) For any $k \in [l]$ and $i \in [n]$, if $\mathcal{W}_k(\mathbf{x}_{k-1,i})$ is independent with the first $d_{\mathcal{L}}$ dimension of the input $\mathbf{x}_{k-1,i}$ then

$$L(\boldsymbol{\theta}_l) = \underbrace{\frac{1}{n}\|\mathcal{P}_{\widehat{\mathbf{D}}}^{\perp}\,\mathrm{vec}\,(\mathbf{Y})\,\|^2}_{\substack{\textit{global optimum value of linear} \\ \textit{regression with basis matrix } \widehat{\mathbf{D}}}},$$

where $\mathbf{D}$ is the same with (i), $\widehat{\mathbf{X}} := \left[\mathbf{X}_1 \otimes \mathbf{I}_{d_y} \quad \mathbf{X}_2 \otimes \mathbf{I}_{d_y} \quad \cdots \quad \mathbf{X}_l \otimes \mathbf{I}_{d_y}\right]^\top$, $\mathbf{X}_k := \left[\mathbf{x}_{k,1} \quad \mathbf{x}_{k,2} \quad \cdots \quad \mathbf{x}_{k,n}\right]$, $\forall k \in [l]$ and $\widehat{\mathbf{D}} := \left[\widehat{\mathbf{X}} \quad \mathbf{D}\right]$.

Theorem 9 is applicable to a wide range of DNNs, ranging from under-parameterized shallow networks to over-parameterized deep architectures. It makes connections between the training objective of MCN and the global minimum value of linear regression, in which the basis matrix is composed of the network parameters and the outputs of hidden layers. When the MCN architecture goes deeper and wider, the column space of $\widehat{\mathbf{D}}$ expands and thus $\mathcal{P}_{\mathbf{D}}^{\perp}\,\mathrm{vec}\,(\mathbf{Y})$ deflates and, accordingly, the training objective may decrease. In other words, for the squared regression problems, the training performance of MCN becomes better as the depth increases even in the worst scenario. So for our MCN, it is the deeper the better.

B.1. Proof of Theorem 9

Proof. Similar to the proof Theorem 1, we simplify MCN as:

$$\mathbf{x}_{k+1,i} = \left[\mathbf{L}_{k+1}\mathbf{x}_{k,i};\ \tilde{\mathbf{A}}_{k+1}\mathbf{x}_i + \max\left\{\mathbf{W}_{k+1}\mathbf{x}_{k,i}, \sigma\left(\mathbf{A}_{k+1}\mathbf{x}_i\right)\right\}\right].$$

In this section, we denote the linear transformation of the output of MCN $\mathbf{Y}_{\boldsymbol{\theta}} := \left[\Psi\left(\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_1)\right), \cdots, \Psi\left(\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_n)\right)\right] \in \mathbb{R}^{d_y \times n}$. Denote the target matrix as $\mathbf{Y} := [\mathbf{y}_1, \cdots, \mathbf{y}_n]$ and the training data as $\mathbf{X} := [\mathbf{x}_1, \cdots, \mathbf{x}_n]$. Denote by $\mathbf{X}_k := [\mathbf{x}_{k,1}, \cdots, \mathbf{x}_{k,n}]$ the output of the k-th layer.

Geven $\boldsymbol{\theta}_k$ as a local minimum of the loss function L, we define several notations. First, we define a mask operator $\hat{\boldsymbol{\Lambda}}_k$, for $(j_1, j_2) \in [d_k] \times [n]$, such that:

$$\left(\hat{\boldsymbol{\Lambda}}_k\right)_{(j_1,j_2)} := \begin{cases} 1, & \text{if} \quad \left(\mathbf{W}_k\mathbf{x}_{k-1,i}\right)_{(j_1,j_2)} \geq \left(\sigma\left(\mathbf{A}_k\mathbf{x}_i\right)\right)_{(j_1,j_2)}; \\ 0, & \text{otherwise.} \end{cases}$$

We also define:

$$\tilde{\boldsymbol{\Lambda}}_k = \operatorname{diag}\left(\begin{bmatrix}\mathbf{1} \\ \operatorname{vec}(\hat{\boldsymbol{\Lambda}}_k)\end{bmatrix}\right),$$

where $\mathbf{1} \in \mathbb{R}^{d_\mathcal{L} n}$ is the all one vector and diag is the diagonal operator. We denote the complementary matrix of $\tilde{\boldsymbol{\Lambda}}_l$ as $\tilde{\boldsymbol{\Lambda}}_l^\perp := \mathbf{1} - \tilde{\boldsymbol{\Lambda}}_l$, where $\mathbf{1}$ is the all one matrix with the compatibility dimension. Let

$$\widetilde{\mathbf{W}}_k = \begin{bmatrix}\mathbf{L}_k \\ \mathbf{W}_k\end{bmatrix}, \quad \mathbf{b}_k = \begin{bmatrix}\mathbf{0} \\ \operatorname{vec}\left(\sigma\left(\mathbf{A}_k\mathbf{X}\right)\right)\end{bmatrix} \quad \text{and} \quad \mathbf{c}_k = \begin{bmatrix}\mathbf{0} \\ \operatorname{vec}\left(\tilde{\mathbf{A}}_k\mathbf{X}\right)\end{bmatrix},$$

where where $\mathbf{0} \in \mathbb{R}^{d_\mathcal{L} n}$ is the all zero vector. Since $\Psi(\cdot)$ is a learnable linear operator, for brevity, we denote $\Psi(\mathbf{x}_{l,i})$ as $\Psi(\mathbf{x}_i)\mathbf{x}_{l,i}$, and let:

$$\mathbf{C}_{l+1} := \begin{bmatrix}\Psi(\mathbf{x}_i) & & \\ & \ddots & \\ & & \Psi(\mathbf{x}_n)\end{bmatrix} \cdot \tilde{\boldsymbol{\Lambda}}_l, \quad \mathbf{C}_{k+1} := \left(\mathbf{I}_n \otimes \widetilde{\mathbf{W}}_{k+1}\right)\tilde{\boldsymbol{\Lambda}}_k,$$

and

$$\mathbf{C}'_{l+1} := \begin{bmatrix}\Psi(\mathbf{x}_i) & & \\ & \ddots & \\ & & \Psi(\mathbf{x}_n)\end{bmatrix} \cdot \tilde{\boldsymbol{\Lambda}}_l^\perp, \quad \mathbf{C}'_{k+1} := \left(\mathbf{I}_n \otimes \widetilde{\mathbf{W}}_{k+1}\right)\tilde{\boldsymbol{\Lambda}}_k^\perp.$$

With these notations, we can have the following two claims.

Claim 11. *For all $k \in [l]$, and,we have:*

$$\partial_{\widetilde{\mathbf{W}}_k}\mathbf{Y}_{\boldsymbol{\theta}} = \mathbf{D}_k,$$

where

$$\mathbf{D}_k = \mathbf{C}_{l+1}\cdots\mathbf{C}_{k+1}\left(\mathbf{X}_{k-1}^\top \otimes \mathbf{I}_{d_k}\right).$$

Proof. We can rewrite MCN as the vectorized form:

$$\begin{aligned}\operatorname{vec}(\mathbf{X}_k) &= \tilde{\boldsymbol{\Lambda}}_k \operatorname{vec}(\widetilde{\mathbf{W}}_k\mathbf{X}_{k-1}) + \tilde{\boldsymbol{\Lambda}}_k^\perp \cdot \operatorname{vec}\left(\begin{bmatrix}\mathbf{0} \\ \sigma\left(\mathbf{A}_k\mathbf{X}\right)\end{bmatrix}\right) + \operatorname{vec}\left(\begin{bmatrix}\mathbf{0} \\ \tilde{\mathbf{A}}_k\mathbf{X}\end{bmatrix}\right) \\ &= \tilde{\boldsymbol{\Lambda}}_k\left(\mathbf{I}_n \otimes \widetilde{\mathbf{W}}_k\right)\operatorname{vec}\left(\mathbf{X}_{k-1}\right) + \tilde{\boldsymbol{\Lambda}}_k^\perp\mathbf{b}_k + \mathbf{c}_k \\ &= \tilde{\boldsymbol{\Lambda}}_k\left(\mathbf{X}_{k-1}^\top \otimes \mathbf{I}_{d_k}\right)\operatorname{vec}\left(\widetilde{\mathbf{W}}_k\right) + \tilde{\boldsymbol{\Lambda}}_k^\perp\mathbf{b}_k + \mathbf{c}_k.\end{aligned}$$

By the definition of $\mathbf{C}$ and $\mathbf{C}'$, we have:

$$\operatorname{vec}(\mathbf{Y}_{\boldsymbol{\theta}}) = \left(\prod_{k'=k}^{\overleftarrow{l}} \mathbf{C}_{k'+1}\right)\left(\mathbf{X}_{k-1}^\top \otimes \mathbf{I}_{d_k}\right)\operatorname{vec}\left(\widetilde{\mathbf{W}}_k\right) + \sum_{j=k}^{l}\left(\prod_{k'=j+2}^{\overleftarrow{l+1}} \mathbf{C}_{k'}\right)\left(\mathbf{C}'_{j+1}\mathbf{b}_j + \mathbf{c}'_j\right), \tag{12}$$

where we let

$$\mathbf{c}'_j = \left(\mathbf{I}_n \otimes \widetilde{\mathbf{W}}_{j+1}\right)\mathbf{c}_j, \quad \overleftarrow{\prod_{k'=l+2}^{l+1}} \mathbf{C}_{k'} = \mathbf{I} \quad \text{and} \quad \overleftarrow{\prod_{k'=k}^{l}} \mathbf{C}_{k'+1} = \mathbf{C}_{l+1}\cdots\mathbf{C}_{k+1}.$$

We finish the proof of this claim. □

Claim 12. *For all $l \in [L]$ and $i \in [n]$, if $\mathcal{W}_l(\mathbf{x}_{k-1,i})$ is independent with the first $d_{\mathcal{L}}$ dimension of the input $\mathbf{x}_{k-1,i}$ and $\Psi(\mathbf{x}_i)$ is a learnable matrix, then we have:*

$$(\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y})\,\mathbf{X}_l^\top = \mathbf{0}.$$

Proof. Since $\mathcal{W}_k(\mathbf{x}_{k-1,i})$ is independent to the first $d_{\mathcal{L}}$ dimension of the input $\mathbf{x}_{k-1,i}$, we can rewrite $\widetilde{\mathbf{W}}_k$ as:

$$\begin{bmatrix} \mathbf{F}_k & \mathbf{G}_k \\ 0 & \mathbf{H}_k \end{bmatrix} := \widetilde{\mathbf{W}}_k.$$

where $\begin{bmatrix}\mathbf{F}_k & \mathbf{G}_k\end{bmatrix} := \mathbf{L}_k$ and $\mathbf{F}_k \in \mathbb{R}^{d_{\mathcal{L}} \times d_{\mathcal{L}}}$. We can have:

$$\begin{aligned}
\mathbf{X}_{k+1} &= \begin{bmatrix} \mathbf{1} \\ \hat{\boldsymbol{\Lambda}}_{k+1} \end{bmatrix} \circ \widetilde{\mathbf{W}}_{k+1} \begin{bmatrix} \mathbf{L}_k\mathbf{X}_{k-1} \\ \overline{\mathbf{X}}_k \end{bmatrix} + \begin{bmatrix} 0 \\ \tilde{\boldsymbol{\Lambda}}^{\perp}_{k+1} \circ (\sigma\left(\mathbf{A}_{k+1}\mathbf{X}\right)) \end{bmatrix} + \begin{bmatrix} 0 \\ \tilde{\mathbf{A}}_{k+1}\mathbf{X} \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{F}_{k+1}\mathbf{L}_k\mathbf{X}_{k-1} + \mathbf{G}_{k+1}\overline{\mathbf{X}}_k \\ \tilde{\boldsymbol{\Lambda}}_{k+1} \circ \mathbf{H}_{k+1}\overline{\mathbf{X}}_k + \tilde{\boldsymbol{\Lambda}}^{\perp}_{k+1} \circ (\sigma\left(\mathbf{A}_{k+1}\mathbf{X}\right)) \end{bmatrix} + \begin{bmatrix} 0 \\ \tilde{\mathbf{A}}_{k+1}\mathbf{X} \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{F}_{k+1}\mathbf{L}_k\mathbf{X}_{k-1} + \mathbf{G}_{k+1}\overline{\mathbf{X}}_k \\ \overline{\mathbf{X}}_{k+1} \end{bmatrix}
\end{aligned},$$

where $\overline{\mathbf{X}}_k$ is the lower $(d_k - d_{\mathcal{L}})$-row part of the matrix $\mathbf{X}_k$. Note that terms $\mathbf{G}_{k+1}\overline{\mathbf{X}}_k$ and $\overline{\mathbf{X}}_{k+1}$ are independent with the learnable matrix $\mathbf{L}_k$.

Without loss of generality, for all $\mathbf{x}_i \in \{\mathbf{x}_i\}_{i=1}^n$, we let $\begin{bmatrix}\mathbf{F}_{k+1} & \mathbf{G}_{k+1}\end{bmatrix} := \Psi(\cdot)$, then we can get:

$$\mathbf{Y}_{\boldsymbol{\theta}} = \left(\overleftarrow{\prod_{l'=k+2}^{l+1}} \mathbf{F}_{l'}\right)\mathbf{L}_{k+1}\mathbf{X}_k + \sum_{j=k}^{l-1}\left(\overleftarrow{\prod_{l'=j+3}^{l+1}} \mathbf{F}_{l'}\right)\mathbf{G}_{k+2}\overline{\mathbf{X}}_{k+1}, \tag{13}$$

where

$$\overleftarrow{\prod_{l'=l+2}^{l+1}} \mathbf{F}_{l'} = \mathbf{I}.$$

Note that:

$$L(\boldsymbol{\theta}) = \frac{1}{n}\|\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y}\|_F^2.$$

By the first order condition of the local minimum, we have:

$$\mathbf{0} = \partial_{\mathbf{L}_{k+1}} L(\boldsymbol{\theta}) = \left(\mathbf{F}_{l+1}\cdots\mathbf{F}_{k+2}\right)^\top (\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y})\,\mathbf{X}_l^\top. \tag{14}$$

If $(\mathbf{F}_{l+1}\cdots\mathbf{F}_{k+2}) \in \mathbb{R}^{d_{\mathcal{L}} \times d_{\mathcal{L}}}$ is full rank, then we finish this proof. Hence, in the rest of this proof, we consider the case:

$$\operatorname{rank}\left(\overleftarrow{\prod_{l'=k+2}^{l+1}} \mathbf{F}_{l'}\right) < d_{\mathcal{L}}.$$

Choosing a unit length vector from the null space of matrix $\left(\mathbf{F}_{l+1}\cdots\mathbf{F}_{k+2}\right)^\top$, i.e.,

$$\|\mathbf{u}_{k+1}\| = 1, \quad \mathbf{u}_{k+1} \in \operatorname{null}\left(\overleftarrow{\prod_{l'=k+2}^{1+1}} \mathbf{F}_{l'}^\top\right) \subset \mathbb{R}^{d_{\mathcal{L}}},$$

where $\mathrm{null}(\cdot)$ denotes the null space of a matrix.

For any $\mathbf{v}_{k+1} \in \mathbb{R}^{d_k}$, we have:

$$\mathbf{Y}_{\boldsymbol{\theta}} = \mathbf{Y}_{\widetilde{\boldsymbol{\theta}}} := \left(\prod_{l'=k+2}^{\overleftarrow{l+1}} \mathbf{F}_{l'} \right) \widetilde{\mathbf{L}}_{k+1} \mathbf{X}_k + \sum_{j=k}^{l-1} \left(\prod_{l'=j+3}^{\overleftarrow{l+1}} \mathbf{F}_{l'} \right) \mathbf{G}_{k+2} \overline{\mathbf{X}}_{k+1},$$

where

$$\widetilde{\mathbf{L}}_{k+1} = \mathbf{L}_{k+1} + \mathbf{u}_{k+1}\mathbf{v}_{k+1}^\top, \quad \widetilde{\boldsymbol{\theta}} = \{\boldsymbol{\theta} \setminus \mathbf{L}_{k+1}, \widetilde{\mathbf{L}}_{k+1}\}.$$

Since $\mathbf{Y}_{\boldsymbol{\theta}} = \mathbf{Y}_{\widetilde{\boldsymbol{\theta}}}$, for any sufficient small $\mathbf{v}_{k+1}$, we can conclude that $\widetilde{\boldsymbol{\theta}}$ is also a local minimum of the loss function L. Similar to the Eq. (14), we have

$$\mathbf{0} = \partial_{\mathbf{F}_{k+1}} L(\widetilde{\boldsymbol{\theta}}) = (\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y}) \, \mathbf{X}_k^\top \widetilde{\mathbf{L}}_{k+1}^\top \left(\prod_{l'=k+2}^{l} \mathbf{F}_{l'}^\top \right).$$

Together with $\mathbf{0} = \partial_{\mathbf{F}_{k+1}} L(\boldsymbol{\theta})$, we can have

$$\mathbf{0} = (\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y}) \, \mathbf{X}_k^\top \left(\mathbf{v}_{k+1}\mathbf{u}_{k+1}^\top \right) \left(\prod_{l'=k+2}^{l} \mathbf{F}_{l'}^\top \right). \tag{15}$$

We now show that,

$$\mathbf{0} = (\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y}) \, \mathbf{X}_k^\top \left(\mathbf{v}_{k+1}\mathbf{u}_{k+1}^\top \right) \left(\prod_{l'=k+2}^{j} \mathbf{F}_{l'}^\top \right), \tag{16}$$

by induction on the index $j = \{l, l-1, \cdots, k+1\}$. The base case $j = l$ is proven above. We consider the case that $j = l - 1$. If $\mathbf{F}_{l+1}$ is full rank, then we have:

$$\mathbf{0} = \partial_{\mathbf{F}_l} L(\widetilde{\boldsymbol{\theta}}) = \mathbf{F}_{l+1}^\top (\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y}) \, \mathbf{X}_k^\top \widetilde{\mathbf{L}}_{k+1}^\top \left(\prod_{l'=k+2}^{l-1} \mathbf{F}_{l'}^\top \right),$$

which indicates that Eq. (16) holds for $j = l - 1$. Now we assume $\mathrm{rank}(\mathbf{F}_{l+1}) < d_{\mathcal{L}}$. Similarly, choosing a unit length vector from the null space of matrix $\mathbf{F}_{l+1}$, i.e.,

$$\|\mathbf{u}_l\| = 1, \quad \mathbf{u}_l \in \mathrm{null}\,(\mathbf{F}_{l+1}) \subset \mathbb{R}^{d_{\mathcal{L}}}.$$

Define:

$$\widetilde{\mathbf{F}}_l = \mathbf{F}_l + \mathbf{u}_l\mathbf{v}_l^\top, \quad \widetilde{\boldsymbol{\theta}}' = \{\widetilde{\boldsymbol{\theta}} \setminus \mathbf{F}_l, \widetilde{\mathbf{F}}_l\},$$

where $\mathbf{v}_l \in \mathbb{R}^{d_{\mathcal{L}}}$. Similarly, we can get $\mathbf{Y}_{\boldsymbol{\theta}} = \mathbf{Y}_{\widetilde{\boldsymbol{\theta}}'}$. Hence, for any sufficient small $\mathbf{v}_l$, we can conclude that $\widetilde{\boldsymbol{\theta}}'$ is also a local minimum of the loss function L, then:

$$\mathbf{0} = \partial_{\mathbf{F}_{l+1}} L(\widetilde{\boldsymbol{\theta}}') = (\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y}) \, \mathbf{X}_k^\top \widetilde{\mathbf{L}}_{k+1}^\top \left(\prod_{l'=k+2}^{l-1} \mathbf{F}_{l'}^\top \right) \widetilde{\mathbf{F}}_l.$$

Together with $\mathbf{0} = \partial_{\mathbf{F}_{l+1}} L(\boldsymbol{\theta})$ and Eq. (15), we can have:

$$\mathbf{0} = (\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y}) \, \mathbf{X}_k^\top \widetilde{\mathbf{L}}_{k+1}^\top \left(\prod_{l'=k+2}^{l} \mathbf{F}_{l'}^\top \right) (\mathbf{v}_l \mathbf{u}_l^\top).$$

Notice that we can easily have

$$\mathbf{0} = (\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y}) \, \mathbf{X}_k^\top \mathbf{L}_{k+1}^\top \left(\prod_{l'=k+2}^{l} \mathbf{F}_{l'}^\top \right) (\mathbf{v}_l \mathbf{u}_l^\top),$$

by setting $\widetilde{\boldsymbol{\theta}}' = \{\boldsymbol{\theta} \setminus \mathbf{F}_l, \widetilde{\mathbf{F}}_l\}$ and using the first order condition w.r.t. the matrix $\mathbf{F}_{l+1}$. Thus, we can conclude:

$$\mathbf{0} = (\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y})\, \mathbf{X}_k^\top \left(\mathbf{v}_{k+1}\mathbf{u}_{k+1}^\top\right) \left(\prod_{l'=k+2}^{l} \mathbf{F}_{l'}^\top\right) (\mathbf{v}_l \mathbf{u}_l^\top),$$

which also implies

$$\mathbf{0} = (\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y})\, \mathbf{X}_k^\top \left(\mathbf{v}_{k+1}\mathbf{u}_{k+1}^\top\right) \left(\prod_{l'=k+2}^{l} \mathbf{F}_{l'}^\top\right) \mathbf{v}_l.$$

The above enquality holds for all sufficient small $\mathbf{v}_l$. We can conclude that Eq. (16) holds for $j = l-1$. This completes the inductive step and proves that:

$$\mathbf{0} = (\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y})\, \mathbf{X}_k^\top \left(\mathbf{v}_{k+1}\mathbf{u}_{k+1}^\top\right),$$

which obviously implies:

$$\mathbf{0} = (\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y})\, \mathbf{X}_k^\top.$$

We now finish the proof of this claim. □

Proof of Theorem 9 (i) From the first order necessary condition of differentiable local minima, we have:

$$\mathbf{0} = \partial_{\widetilde{\mathbf{W}}_k} L(\boldsymbol{\theta}) = \mathbf{D}_k^\top \operatorname{vec}(\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y}),$$

where the last equation comes from Claim 11. Let

$$\mathbf{D} := \begin{bmatrix} \mathbf{D}_1 & \mathbf{D}_2 & \cdots & \mathbf{D}_{l+1} \end{bmatrix}.$$

We have

$$\mathbf{0} = \mathbf{D}^\top \operatorname{vec}(\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y}).$$

According to the Eq. (12), it is obvious that $\operatorname{vec}(\mathbf{Y}_{\boldsymbol{\theta}})$ belongs to the column space of matrix $\mathbf{D}$. Thus, we can conclude:

$$\operatorname{vec}(\mathbf{Y}_{\boldsymbol{\theta}}) = \mathcal{P}_{\mathbf{D}} \operatorname{vec}(\mathbf{Y}).$$

Therefore,

$$nL(\boldsymbol{\theta}) = \|\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y}\|_F^2 = \|\operatorname{vec}(\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y})\|^2 = \|\mathcal{P}_{\mathbf{D}} \operatorname{vec}(\mathbf{Y}) - \operatorname{vec}(\mathbf{Y})\|^2 = \|\mathcal{P}_{\mathbf{D}}^\perp \operatorname{vec}(\mathbf{Y})\|^2.$$

Proof of Theorem 9 (ii) From Claim 12, we have:

$$\mathbf{0} = \left(\mathbf{X}_k \otimes \mathbf{I}_{d_y}\right) \operatorname{vec}(\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y}),$$

Let

$$\widehat{\mathbf{X}} := \begin{bmatrix} \mathbf{X}_1 \otimes \mathbf{I}_{d_y} & \mathbf{X}_2 \otimes \mathbf{I}_{d_y} & \cdots & \mathbf{X}_l \otimes \mathbf{I}_{d_y} \end{bmatrix}^\top$$

We have

$$\mathbf{0} = \widehat{\mathbf{X}}^\top \operatorname{vec}(\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y}).$$

Combine the result of Claim 11, we can get:

$$\mathbf{0} = \begin{bmatrix} \widehat{\mathbf{X}} & \mathbf{D} \end{bmatrix}^\top \operatorname{vec}(\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y}).$$

According to the Eq. (13), it is obvious that $\operatorname{vec}(\mathbf{Y}_{\boldsymbol{\theta}})$ belongs to the column space of matrix $\widehat{\mathbf{X}}$. Thus, we can conclude:

$$\operatorname{vec}(\mathbf{Y}_{\boldsymbol{\theta}}) = \mathcal{P}_{\begin{bmatrix} \widehat{\mathbf{X}} & \mathbf{D} \end{bmatrix}} \operatorname{vec}(\mathbf{Y}).$$

Therefore,

$$\begin{aligned} nL(\boldsymbol{\theta}) &= \|\operatorname{vec}(\mathbf{Y}_{\boldsymbol{\theta}} - \mathbf{Y})\|^2 = \|\mathcal{P}_{\begin{bmatrix} \widehat{\mathbf{X}} & \mathbf{D} \end{bmatrix}} \operatorname{vec}(\mathbf{Y}) - \operatorname{vec}(\mathbf{Y})\|^2 \\ &= \|\mathcal{P}_{\widehat{\mathbf{X}}} \operatorname{vec}(\mathbf{Y}) - \operatorname{vec}(\mathbf{Y}) + \mathcal{P}_{[\mathcal{P}_{\widehat{\mathbf{X}}}^\perp \cdot \mathbf{D}]} \operatorname{vec}(\mathbf{Y})\|^2 \\ &= \|\mathcal{P}_{[\mathcal{P}_{\widehat{\mathbf{X}}}^\perp \cdot \mathbf{D}]} \operatorname{vec}(\mathbf{Y}) - \mathcal{P}_{\widehat{\mathbf{X}}}^\perp \operatorname{vec}(\mathbf{Y})\|^2 \\ &= \|\mathcal{P}_{\widehat{\mathbf{X}}}^\perp \operatorname{vec}(\mathbf{Y})\|^2 - \|\mathcal{P}_{[\mathcal{P}_{\widehat{\mathbf{X}}}^\perp \cdot \mathbf{D}]} \operatorname{vec}(\mathbf{Y})\|^2. \end{aligned}$$

We now finish the whole proof. □

C. Prior Arts

C.1. Effects of Depth and Width in Neural Networks

Usually, each layer of wide networks contains abundant hidden units, and these units can be seen as one kind of features. Hence, wide networks (even infinitely wide) naturally have connection with the kernels and Gaussian processes. By the kernel methods, the works in (Xie et al., 2017; Du et al., 2019b) lower bounded the spectrum of Gram matrix and revealed that the network learning is actually a regression problem, but their theoretical bounds only hold for shallow networks. Then works (Du et al., 2019a; Arora et al., 2019b) captured the behavior of fully-connected deep networks in the large (maybe infinite) width limit trained by gradient descent and also found the equivalence between the kernel regression predictor and wide networks. However, all these works do not show the benefits of depth, and deeper nets do not obtain better theoretical results than shallow ones in their settings.

Depth is also important to the general networks. Generally, a neural network with $\Theta(k^3)$ layers, $\Theta(1)$ units per layer, cannot be approximated by networks with $O(k)$ layers (Telgarsky, 2016). The works (Kawaguchi et al., 2019; Arora et al., 2018) showed that deeper and wider fully-connected networks obtain better training results, but did not analyze the NN's performance during testing. By contrast, besides showing the training objective decreases monotonously with the increase of depth and width, we also give the generalization bound of the proposed MCN. In addition, we prove that $(l+1)$-layer MCN always obtains better training results than l-layer MCN, which reveals the reason why deeper nets usually perform better in practice.

C.2. Generalization of Neural Networks

One major concern in the learning community is the generalization bound (also known as estimation bound). In general, at least $n = \Omega(\epsilon^{-\max\{d_x,2\}})$ samples are needed to learn a Lipschitz-continuous functions in $\mathbb{R}^{d_x}$ with the population regression risk as ϵ (Luxburg & Bousquet, 2004). The exponential dependence on the dimension d_x is often referred to as the *curse of dimensionality*. Fortunately, when the model structure is specified, the sample complexity can be reduced, e.g., $\Omega(d_x\epsilon^{-2})$ for affine functions (Shalev-Shwartz & Ben-David, 2014), $\Omega(k^2 d_x \epsilon^{-2})$ for single hidden-layer fully connected neural networks (Rumerlhar, 1986), where k is the number of units in the hidden layer, and $\widetilde{O}\left(m^2\epsilon^{-4}\right)$ for one-hidden-layer CNN with m-dimensional convolutional filter (Du et al., 2018). Generally, for a parametric regression problem, the expected generalization error is bounded as $O(D\log(n)/n)$, where D is depends on the amount of model parameters (Maillard & Munos, 2009; Györfi et al., 2006). Obviously, this bound cannot reveal the mystery of generalization ability of over-parametrized deep learning models which have more parameters than necessary to fit the training data.

In practice, we first train DNNs to perfectly fit the training data. The resulting (zero training loss) NNs can already have good performance on test data (Zhang et al., 2017). This phenomena is considered as one of reasons to concern the theoretical generalization bound of neural networks. Some researchers try to find the inspiration from shallow networks. By assuming the existence of a true model, the works in (Ma et al., 2018; Du et al., 2018; Arora et al., 2019c) showed that the (regularized) empirical risk minimizer has good generalization with sample complexity that depends on the true model. Another line of researchers take the dynamic optimization process (e.g., SGD) into consideration and/or connect the network learning with kernel methods (Arora et al., 2019c; Allen-Zhu et al., 2018; Dou & Liang, 2019). Although the theory is rigorous, all the works cannot be easily extended to the networks with complex structure which may not be trained by SGD.

Surprisingly, some recent works found that data interpolation also have good generalization ability and even can obtain the statistical sub-optimality and optimality for linear and kernel-based combination of observation, respectively (Belkin et al., 2018b; 2019). Moreover, bias-variance trade-off theory for interpolating predictors was also explored (Belkin et al., 2018a). However, all these works are non-parametric and may not directly apply to the DNN analysis.