Deep Frequency Principle Towards Understanding Why Deeper Learning Is Faster

Zhiqin John Xu and Hanxu Zhou

School of Mathematical Sciences, MOE-LSC and Institute of Natural Sciences, Shanghai Jiao Tong University, Shanghai, 200240, P.R. China

xuzhiqin@sjtu.edu.cn, 1210123652@sjtu.edu.cn

Abstract

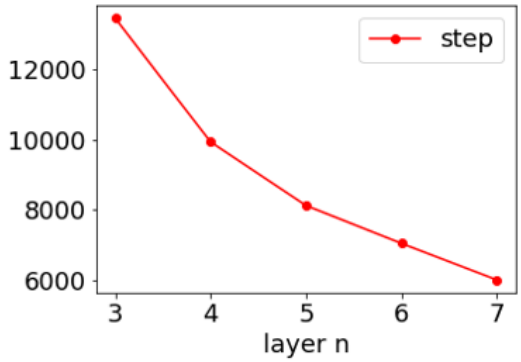
Understanding the effect of depth in deep learning is a critical problem. In this work, we utilize the Fourier analysis to empirically provide a promising mechanism to understand why feedforward deeper learning is faster. To this end, we separate a deep neural network, trained by normal stochastic gradient descent, into two parts during analysis, i.e., a pre-condition component and a learning component, in which the output of the pre-condition one is the input of the learning one. We use a filtering method to characterize the frequency distribution of a high-dimensional function. Based on experiments of deep networks and real dataset, we propose a deep frequency principle, that is, the effective target function for a deeper hidden layer biases towards lower frequency during the training. Therefore, the learning component effectively learns a lower frequency function if the pre-condition component has more layers. Due to the well-studied frequency principle, i.e., deep neural networks learn lower frequency functions faster, the deep frequency principle provides a reasonable explanation to why deeper learning is faster. We believe these empirical studies would be valuable for future theoretical studies of the effect of depth in deep learning.

Introduction

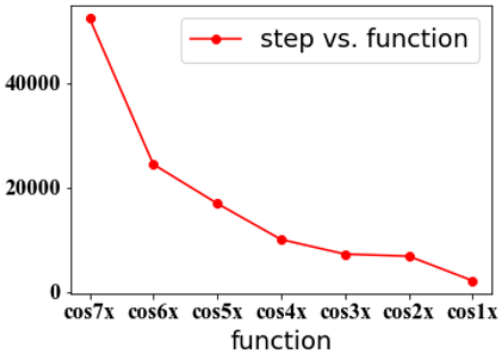
Deep neural networks have achieved tremendous success in many applications, such as computer vision, speech recognition, speech translation, and natural language processing etc. The depth in neural networks plays an important role in the applications. Understanding the effect of depth is a central problem to reveal the "black box" of deep learning. For example, empirical studies show that a deeper network can learn faster and generalize better in both real data and synthetic data (He et al. 2016; Arora, Cohen, and Hazan 2018). Different network structures have different computation costs in each training epoch. In this work, we define that *the learning of a deep neural network is faster if the loss of the deep neural network decreases to a designated error with fewer training epochs.* For example, as shown in Fig. 1 (a), when learning data sampled from a target function $\cos(3x) + \cos(5x)$, a deep neural network with more hidden layers achieves the designated training loss with fewer training epochs. Although empirical studies suggest deeper neural networks may learn faster, there is few understanding of the mechanism.

In this work, we would empirically explore an underlying mechanism that may explain why deeper neural network (note: in this work, we only study feedforward networks) can learn faster from the perspective of Fourier analysis. We start from a universal phenomenon of frequency principle (Xu, Zhang, and Xiao 2019; Rahaman et al. 2019; Xu et al. 2020; Luo et al. 2019; E, Ma, and Wu 2020), that is, deep neural networks often fit target functions from low to high frequencies during the training. Recent works show that frequency principle may provide an understanding to the success and failure of deep learning (Xu et al. 2020; Zhang et al. 2019; E, Ma, and Wu 2020; Ma, Wu, and E 2020). We use an ideal example to illustrate the frequency principle, i.e., using a deep neural network to fit different target functions. As the frequency of the target function decreases, the deep neural network achieves a designated error with fewer training epochs, as shown in Fig. 1 (b), which is similar to the phenomenon when using a deeper network to learn a fixed target function.

Inspired by the above analysis, we propose a mechanism to understand why a deeper network, $f_\theta(\boldsymbol{x})$, faster learns a set of training data, $S = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ sampled from a target function $f^*(\boldsymbol{x})$, illustrated as follows. Networks are trained as usual while we separate a deep neural network into two parts in the analysis, as shown in Fig. 2, one is a pre-condition component and the other is a learning component, in which the output of the pre-condition one, denoted as $f_{\boldsymbol{\theta}}^{[l-1]}(\boldsymbol{x})$ (first $l-1$ layers are classified as the pre-condition component), is the input of the learning one. For the learning component, the effective training data at each training epoch is $S^{[l-1]} = \{(f_{\boldsymbol{\theta}}^{[l-1]}(\boldsymbol{x}_i), y_i)\}_{i=1}^n$. We then perform experiments based on the variants of Resnet18 structure (He et al. 2016) and CIFAR10 dataset. We fix the learning component (fully-connected layers). When increasing the number of the pre-condition layer (convolution layers), we find that $S^{[l-1]}$ has a stronger bias towards low frequency during the training. By frequency principle, the learning of a lower frequency function is faster, therefore, the learning component is faster to learn $S^{[l-1]}$ when the pre-condition component has more layers. The analysis among different network structures is often much more difficult than the analy-



(a) different networks



(b) different target functions

Figure 1: Training epochs (indicated by ordinate axis) of different deep neural networks when they achieve a fixed error. (a) Using networks with different number of hidden layers with the same size to learn data sampled from a target function $\cos(3x) + \cos(5x)$. (b) Using a fixed network to learn data sampled from different target functions.

sis of one single structure. For providing hints for future theoretical study, we study a fixed fully-connected deep neural network by classifying different number of layers into the pre-condition component, i.e., varying l for a network in the analysis. As l increases, we similarly find that $S^{[l]}$ contains more low frequency and less high frequency during the training. Therefore, we propose the following principle:

Deep frequency principle: The effective target function for a deeper hidden layer biases towards lower frequency during the training.

With the well-studied frequency principle, the deep frequency principle shows a promising mechanism for understanding why a deeper network learns faster.

Related Work

From the perspective of approximation, the expressive power of a deep neural network increases with the depth (Telgarsky 2016; Eldan and Shamir 2016; E and Qingcan 2018). However, the approximation theory renders no implication on the optimization of deep neural networks.

With residual connection, He et al. (2016) successfully train very deep networks and find that deeper networks can achieve better generalization error. In addition, He et al. (2016) also show that the training of deeper network is faster. Arora, Cohen, and Hazan (2018) show that the acceleration effect of depth also exists in deep linear neural network and provide a viewpoint for understanding the effect of depth, that is, increasing depth can be seen as an acceleration procedure that combines momentum with adaptive learning rates. There are also many works studying the effect of depth for deep linear networks (Saxe, Mcclelland, and Ganguli 2014; Kawaguchi, Huang, and Kaelbling 2019; Gissin, Shalev-Shwartz, and Daniely 2019; Shin 2019). In this work, we study the optimization effect of depth in nonlinear deep networks.

Various studies suggest that the function learned by the deep neural networks increases its complexity as the training goes (Arpit et al. 2017; Valle-Perez, Camargo, and Louis 2018; Mingard et al. 2019; Kalimeris et al. 2019; Yang and Salman 2019). This increasing complexity is also found in deep linear network (Gissin, Shalev-Shwartz, and Daniely 2019). The high-dimensional experiments in (Xu et al. 2020) show that the low-frequency part is converged first, i.e., frequency principle. Therefore, the ratio of the power of the low-frequency component of the deep neural network output experiences a increasing stage at the beginning (due to the convergence of low-frequency part), followed by a decreasing stage (due to the convergence of high-frequency part). As more high-frequency involved, the complexity of the deep neural network output increases. Therefore, the ratio of the low-frequency component used in this paper validates the complexity increasing during the training, which is consistent with other studies.

Frequency principle is examined in extensive datasets and deep neural networks (Xu, Zhang, and Xiao 2019; Rahaman et al. 2019; Xu et al. 2020). Theoretical studies subsequently shows that frequency principle holds in general setting with infinite samples (Luo et al. 2019) and in the regime of wide neural networks (Neural Tangent Kernel (NTK) regime (Jacot, Gabriel, and Hongler 2018)) with finite samples (Zhang et al. 2019) or sufficient many samples (Cao et al. 2019; Yang and Salman 2019; Ronen et al. 2019; Bordelon, Canatar, and Pehlevan 2020). E, Ma, and Wu (2020) show that the integral equation would naturally leads to the frequency principle. With the theoretical understanding, the frequency principle inspires the design of deep neural networks to fast learn a function with high frequency (Liu, Cai, and Xu 2020; Wang et al. 2020; Jagtap, Kawaguchi, and Karniadakis 2020; Cai, Li, and Liu 2019; Biland et al. 2019; Li, Xu, and Zhang 2020).

Preliminary

Low Frequency Ratio (LFR)

To compare two 1-d functions in the frequency domain, we can display their spectrum. However, this does not apply for high-dimensional functions, because the computation cost of high-dimensional Fourier transform suffers from

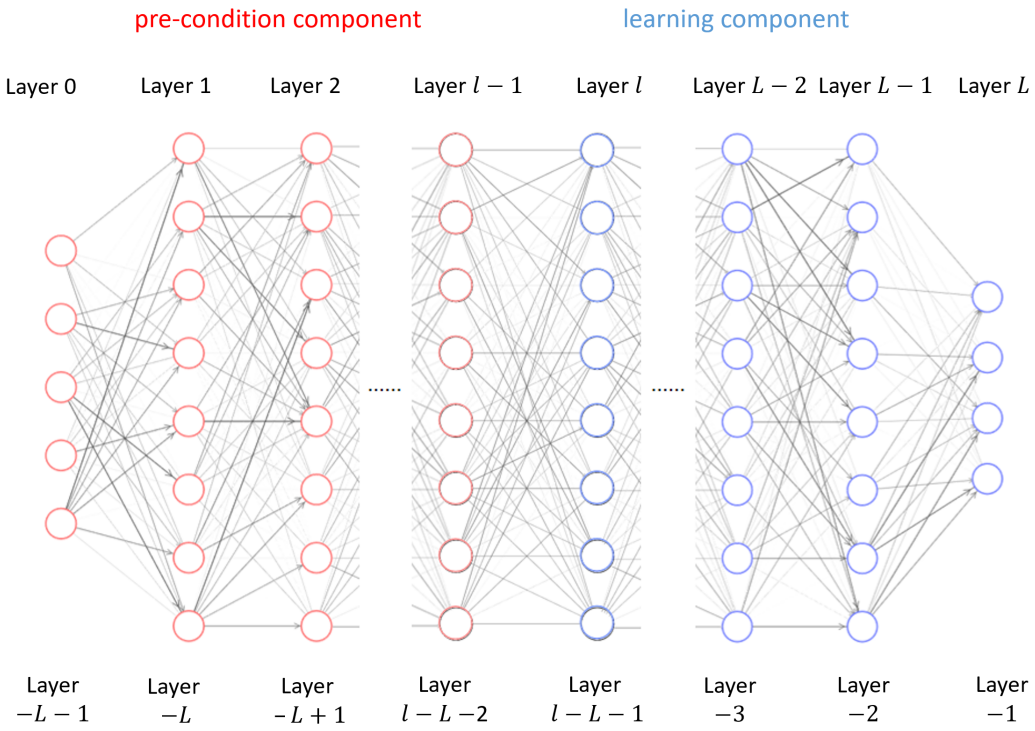


Figure 2: General deep neural network.

the curse of dimensionality. To overcome this, we use a low-frequency filter to derive a low-frequency component of the interested function and then use a Low Frequency Ratio (LFR) to characterize the power ratio of the low-frequency component over the whole spectrum.

The LFR is defined as follows. We first split the frequency domain into two parts, i.e., a low-frequency part with frequency $|\boldsymbol{k}| \leq k_0$ and a high-frequency part with $|\boldsymbol{k}| > k_0$, where $|\cdot|$ is the length of a vector. Consider a dataset $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^n$, $\boldsymbol{x}_i \in R^d$, and $\boldsymbol{y}_i \in R^{d_o}$. For example, $d = 784$ and $d_o = 10$ for MNIST and $d = 3072$ and $d_o = 10$ for CIFAR10. The LFR is defined as

$$\mathrm{LFR}(k_0) = \frac{\sum_{\boldsymbol{k}} \mathbb{1}_{|\boldsymbol{k}|\leq k_0} |\hat{\boldsymbol{y}}(\boldsymbol{k})|^2}{\sum_{\boldsymbol{k}} |\hat{\boldsymbol{y}}(\boldsymbol{k})|^2}, \tag{1}$$

where $\hat{\cdot}$ indicates Fourier transform, $\mathbb{1}_{\boldsymbol{k}\leq k_0}$ is an indicator function, i.e.,

$$\mathbb{1}_{|\boldsymbol{k}|\leq k_0} = \begin{cases} 1, & |\boldsymbol{k}| \leq k_0, \\ 0, & |\boldsymbol{k}| > k_0. \end{cases}$$

However, it is almost impossible to compute above quantities numerically due to high computational cost of high-dimensional Fourier transform. Similarly as previous study (Xu et al. 2020), We alternatively use the Fourier transform of a Gaussian function $\hat{G}^\delta(\boldsymbol{k})$, where δ is the variance of the Gaussian function G, to approximate $\mathbb{1}_{|\boldsymbol{k}|>k_0}$. Note that $1/\delta$ can be interpreted as the variance of $\hat{G}$. The approximation is reasonable due to the following two reasons. First, the Fourier transform of a Gaussian is still a Gaussian, i.e., $\hat{G}^\delta(\boldsymbol{k})$ decays exponentially as $|\boldsymbol{k}|$ increases, therefore, it can approximate $\mathbb{1}_{|\boldsymbol{k}|\leq k_0}$ by $\hat{G}^\delta(\boldsymbol{k})$ with a proper $\delta(k_0)$. Second, the computation of LFR contains the multiplication of Fourier transforms in the frequency domain, which is equivalent to the Fourier transform of a convolution in the spatial domain. We can equivalently perform the computation in the spatial domain so as to avoid the almost impossible high-dimensional Fourier transform. The low frequency part can be derived by

$$\boldsymbol{y}_i^{\mathrm{low},\delta(k_0)} \triangleq (\boldsymbol{y} * G^{\delta(k_0)})_i, \tag{2}$$

where $*$ indicates convolution operator. Then, we can compute the LFR by

$$\mathrm{LFR}(k_0) = \frac{\sum_i |\boldsymbol{y}_i^{\mathrm{low},\delta(k_0)}|^2}{\sum_i |\boldsymbol{y}_i|^2}. \tag{3}$$

The low frequency part can be derived on the discrete data points by

$$\boldsymbol{y}_i^{\mathrm{low},\delta} = \frac{1}{C_i} \sum_{j=0}^{n-1} \boldsymbol{y}_j G^\delta(\boldsymbol{x}_i - \boldsymbol{x}_j), \tag{4}$$

where $C_i = \sum_{j=0}^{n-1} G^\delta(\boldsymbol{x}_i - \boldsymbol{x}_j)$ is a normalization factor and

$$G^\delta(\boldsymbol{x}_i - \boldsymbol{x}_j) = \exp\left(-|\boldsymbol{x}_i - \boldsymbol{x}_j|^2/2\delta\right). \tag{5}$$

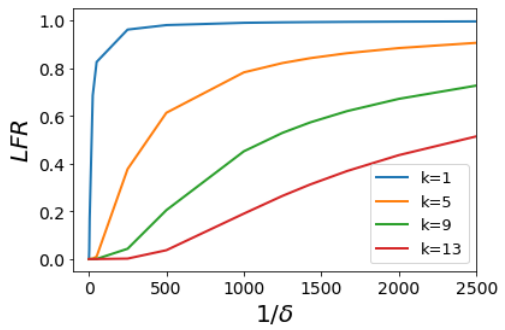
$1/\delta$ is the variance of $\hat{G}$, therefore, it can be interpreted as the frequency width outside which is filtered out by convolution.

Ratio Density Function (RDF)

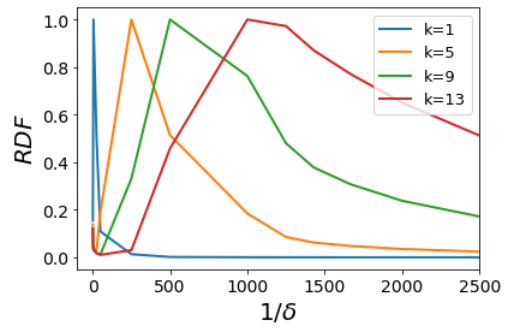
$\mathrm{LFR}(k_0)$ characterizes the power ratio of frequencies within a sphere of radius k_0. To characterize each frequency in the radius direction, similarly to probability, we define the ratio density function (RDF) as

$$\mathrm{RDF}(k_0) = \frac{\partial \mathrm{LFR}(k_0)}{\partial k_0}. \tag{6}$$

In practical computation, we use $1/\delta$ for k_0 and use the linear slope between two consecutive points for the derivative. For illustration, we show the LFR and RDF for $\sin(k\pi x)$ in Fig. 3. As shown in Fig. 3(a), the LFR of low-frequency function faster approaches one when the filter width in the frequency domain is small, i.e., small $1/\delta$. The RDF in Fig. 3(b) shows that as k in the target function increases, the peak of RDF moves towards wider filter width, i.e., higher frequency. Therefore, it is more intuitive that the RDF effectively reflects where the power of the function concentrates in the frequency domain. In the following, we will use RDF to study the frequency distribution of effective target functions for hidden layers.



(a)



(b)

Figure 3: LFR and RDF for $\sin(k\pi x)$ vs. $1/\delta$. Note that we normalize RDF in (b) by the maximal value of each curve for visualization.

General Deep Neural Network

We adopt the suggested standard notation in (BAAI 2020)[1]. An L-layer neural network is defined recursively,

$$f_{\boldsymbol{\theta}}^{[0]}(\boldsymbol{x}) = \boldsymbol{x}, \tag{7}$$

$$f_{\boldsymbol{\theta}}^{[l]}(\boldsymbol{x}) = \sigma \circ (\boldsymbol{W}^{[l-1]} f_{\boldsymbol{\theta}}^{[l-1]}(\boldsymbol{x}) + \boldsymbol{b}^{[l-1]}) \quad 1 \le l \le L-1, \tag{8}$$

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = f_{\boldsymbol{\theta}}^{[L]}(\boldsymbol{x}) = \boldsymbol{W}^{[L-1]} f_{\boldsymbol{\theta}}^{[L-1]}(\boldsymbol{x}) + \boldsymbol{b}^{[L-1]}, \tag{9}$$

where $\boldsymbol{W}^{[l]} \in \mathbb{R}^{m_{l+1} \times m_l}$, $\boldsymbol{b}^{[l]} = \mathbb{R}^{m_{l+1}}$, $m_0 = d_{\mathrm{in}} = d$, $m_L = d_{\mathrm{o}}$, σ is a scalar function and "$\circ$" means entry-wise operation. We denote the set of parameters by $\boldsymbol{\theta}$. For simplicity, we also denote

$$f_{\boldsymbol{\theta}}^{[-l]}(\boldsymbol{x}) = f_{\boldsymbol{\theta}}^{[L-l+1]}(\boldsymbol{x}). \tag{10}$$

For example, the output layer is layer "-1", i.e., $f_{\boldsymbol{\theta}}^{[-1]}(\boldsymbol{x})$ for a given input $\boldsymbol{x}$, and the last hidden layer is layer "-2", i.e., $f_{\boldsymbol{\theta}}^{[-2]}(\boldsymbol{x})$ for a given input $\boldsymbol{x}$, illustrated in Fig. 2.

The effective target function for the learning component, consisting from layer "l" to the output layer, is

$$S^{[l-1]} = \{(f_{\boldsymbol{\theta}}^{[l-1]}(\boldsymbol{x}_i), y_i)\}_{i=1}^n. \tag{11}$$

Training Details

We list training details for experiments as follows.

For the experiments of the variants of Resnet18 on CIFAR10, the network structures are shown in Fig. 4. The output layer is equipped with softmax and the network is trained by Adam optimizer with cross-entropy loss and batch size 256. The learning rate is changed as the training proceeds, that is, 10^{-3} for epoch 1-40 , 10^{-4} for epoch 41-60, and 10^{-5} for epoch 61-80. We use 40000 samples of CIFAR10 as the training set and 10000 examples as the validation set. The training accuracy and the validation accuracy are shown in Fig. 5. The RDF of the effective target function of the last hidden layer for each variant is shown in Fig. 6.

For the experiment of fully-connected network on MNIST, we choose the activation function of $\tanh$ and size $784 - 500 - 500 - 500 - 500 - 500 - 10$. The output layer of the network does not equip any activation function. The network is trained by Adam optimizer with mean squared loss, batch size 256 and learning rate 10^{-5}. The training is stopped when the loss is smaller than 10^{-2}. We use 30000 samples of the MNIST as training set. The RDF of the effective target functions of different hidden layers are shown in Fig. 8.

Note that ranges of different dimensions in the input are different, which would result in that for the same δ, different dimensions keep different frequency ranges when convolving with the Gaussian function. Therefore, we normalized each dimension by its maximum amplitude, thus, each dimension lies in $[-1, 1]$. Without doing such normalization, we still obtain similar results of deep frequency principle.

[1]BAAI.2020. Suggested Notation for Machine Learn-ing. https://github.com/Mayuyu/suggested-notation-for-machine-learning.

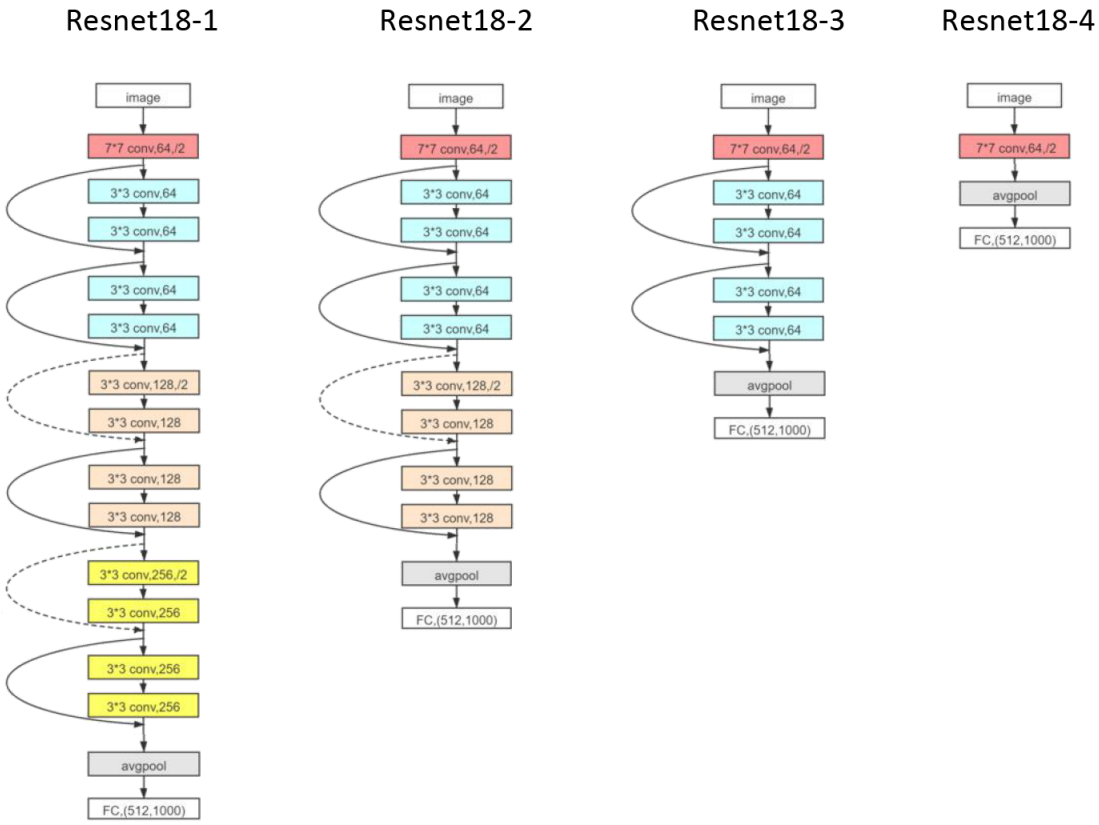
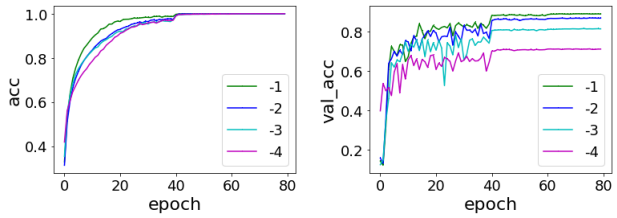


Figure 4: Variants of Resnet18.

All codes are written by Python and Tensorflow, and run on Linux system with Nvidia GTX 2080Ti or Tesla V100 cards. Codes can be found at github.com.



(a) Training (b) Validation

Figure 5: Training accuracy and validation accuracy vs. epoch for variants of Resnet18.

Results

Based on the experiments of deep networks and real datasets, we would show a deep frequency principle, a promising mechanism, to understand why deeper neural networks learn faster, that is, the effective target function for a deeper hidden layer biases towards lower frequency during the training. To derive the effective function, we decompose the target function into a pre-condition component, consisting of layers before the considered hidden layer, and a learning component, consisting from the considered hidden layer to the output layer, as shown in Fig. 2. As the considered hidden layer gets deeper, the learning component effectively learns a lower frequency function. Due to the frequency principle, i.e., deep neural networks learn low frequency faster, a deeper neural network can learn the target function faster. The key to validate deep frequency principle is to show the frequency distribution of the effective target function for each considered hidden layer.

First, we study a practical and common situation, that is, networks with more hidden layers learn faster. Then, we examine the deep frequency principle on a fixed deep neural network but consider the effective target function for different hidden layers.

Deep Frequency Principle on the Variants of Resnet18

In this subsection, we would utilize variants of Resnet18 and CIFAR10 dataset to validate deep frequency principle. The structures of four variants are illustrated as follows. As shown in Fig. 4, all structures have several convolution parts, followed by two same fully-connected layers. Compared with Resnet18-i, Resnet18-$(i+1)$ drops out a convolution part and keep other parts the same.

As shown in Fig. 5, a deeper net attains a fixed training accuracy with fewer training epochs and achieves a better generalization after training.

From the layer "-2" to the final output, it can be regarded as a two-layer neural network, which is widely studied. Next, we examine the RDF for layer "-2". The effective target function is

$$S^{[-3]} = \left\{ \left(f_{\boldsymbol{\theta}}^{[-3]}(\boldsymbol{x}_i), \boldsymbol{y}_i \right) \right\}_{i=1}^{n}. \tag{12}$$

As shown in Fig. 6(a), at initialization, the RDFs for deeper networks concentrate at higher frequencies. However, as training proceeds, the concentration of RDFs of deeper networks moves towards lower frequency faster. Therefore, for the two-layer neural network with a deeper pre-condition component, learning can be accelerated due to the fast convergence of low frequency in neural network dynamics, i.e., frequency principle.

For the two-layer neural network embedded as the learning component of the full network, the effective target function is $S^{[-3]}$. As the pre-condition component has more layers, layer "-2" is a *deeper* hidden layer in the full network. Therefore, Fig. 6 validates that the effective target function for a deeper hidden layer biases towards lower frequency during the training, i.e., deep frequency principle. One may curious about how is the frequency distribution of the effective function of the learning component, i.e., $\{f_{\boldsymbol{\theta}}^{[-3]}(\boldsymbol{x}), f_{\boldsymbol{\theta}}^{[-1]}(\boldsymbol{x})\}$. We consider RDF for the effective function evaluated on training points, that is, $\left\{ \left(f_{\boldsymbol{\theta}}^{[-3]}(\boldsymbol{x}_i), f_{\boldsymbol{\theta}}^{[-1]}(\boldsymbol{x}_i) \right) \right\}_{i=1}^{n}$. This is similar as the effective target function, that is, those in deeper networks bias towards more low frequency function, as shown in Fig. 7.

We have also performed many other experiments and validate the deep frequency principle, such as networks with different activation functions, fully-connected networks without residual connection, and different loss functions. An experiment for discussing the residual connection is presented in the discussion part.

The comparison in experiments above crosses different networks, which would be difficult for future analysis. Alternatively, we can study how RDFs of $S^{[l]}$ of different l in a fixed deep network evolves during the training process. As expected, as l increases, $S^{[l]}$ would be dominated by more lower frequency during the training process.

RDF of Different Hidden Layers in a Fully Connected Deep Neural Network

As analyzed above, an examination of the deep frequency principle in a deep neural network would provide valuable insights for future theoretical study. A key problem is that different hidden layers often have different sizes, i.e., $S^{[l]}$'s have different input dimensions over different l's. LFR is similar to a volume ratio, thus, it depends on the dimension. To control the dimension variable, we consider a fully-connected deep neural network with the same size for different hidden layers to learn MNIST dataset.

As shown in Fig. 8, at initialization, the peak of RDF for a deeper hidden layer locates at a higher frequency. As the training goes, the peak of RDF of a deeper hidden layer moves towards low frequency faster. At the end of the training, the frequency of the RDF peak monotonically decreases as the hidden layer gets deeper. This indicates that the effective target function for a deeper hidden layer evolves faster towards a low frequency function, i.e., the deep frequency principle in a deep neural network.

Discussion

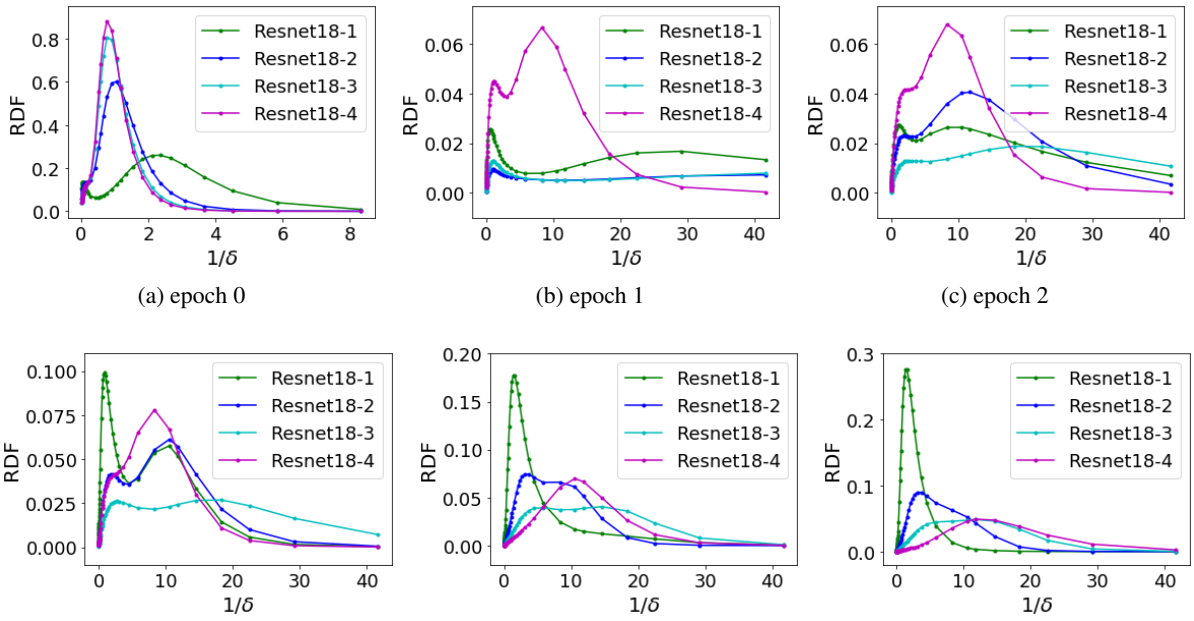
In this work, we empirically show a deep frequency principle that provides a promising mechanism for understanding the effect of depth in deep learning, that is, the effective target function for a deeper hidden layer biases towards lower frequency during the training. Specifically, based on the well-studied frequency principle, the deep frequency principle well explains why deeper learning can be faster. We believe that the study of deep frequency principle would provide valuable insight for further theoretical understanding of deep neural networks. Next, we will discuss the relation of this work to other studies and some implications.

Kernel Methods

Kernel methods, such as support vector machine and random feature model, are powerful at fitting non-linearly separable data. An intuitive explanation is that when data are projected into a much higher dimensional space, they are closer to be linearly separable. From the perspective of Fourier analysis, we quantify this intuition through the low-frequency ratio. After projecting data to higher dimensional space through the hidden layers, neural networks transform the high-dimensional target function into a lower frequency effective function. The deeper neural networks project data more than once into high dimensional space, which is equivalent to the combination of multiple kernel methods. In addition, neural networks not only learn the weights of kernels but also are able to learn the kernels, showing a much more capability compared with kernel methods.

Generalization

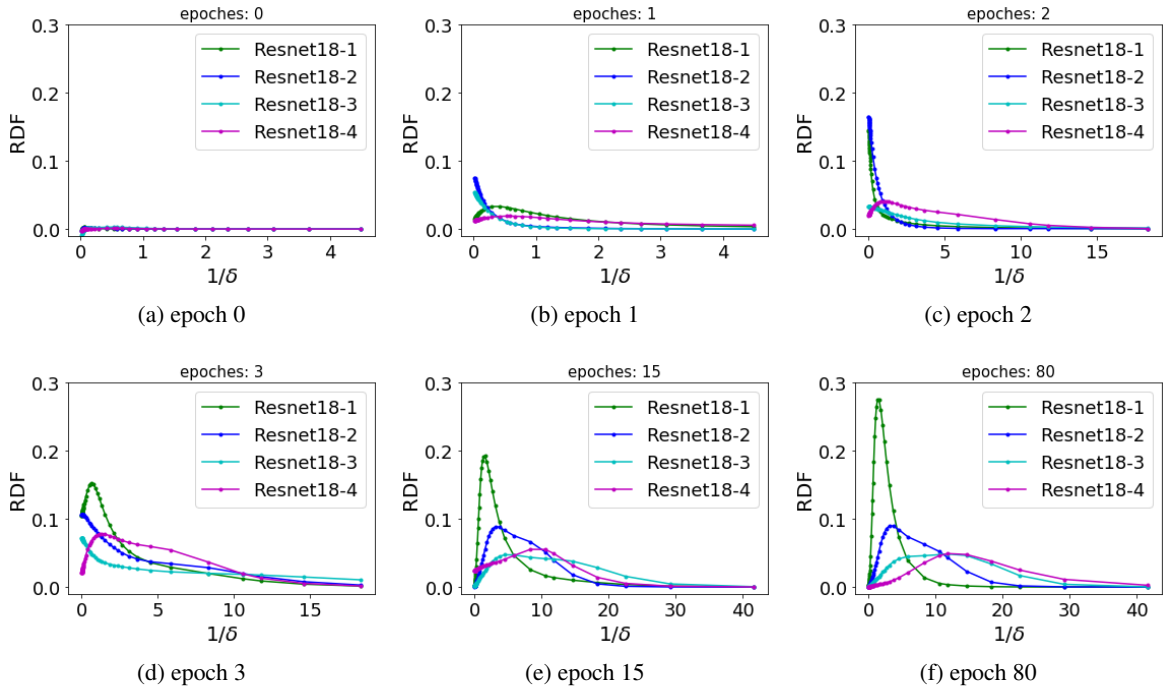
Frequency principle reveals a low-frequency bias of deep neural networks (Xu, Zhang, and Xiao 2019; Xu et al. 2020), which provides qualitative understandings (Xu 2018; Zhang et al. 2019) for the good generalization of neural networks in problems dominated by low frequencies, such as natural image classification tasks, and for the poor generalization in problems dominated by high frequencies, such as predicting parity function. Generalization in real world problems (He et al. 2016) is often better as the network goes deeper.



(a) epoch 0 (b) epoch 1 (c) epoch 2

(d) epoch 3 (e) epoch 15 (f) epoch 80

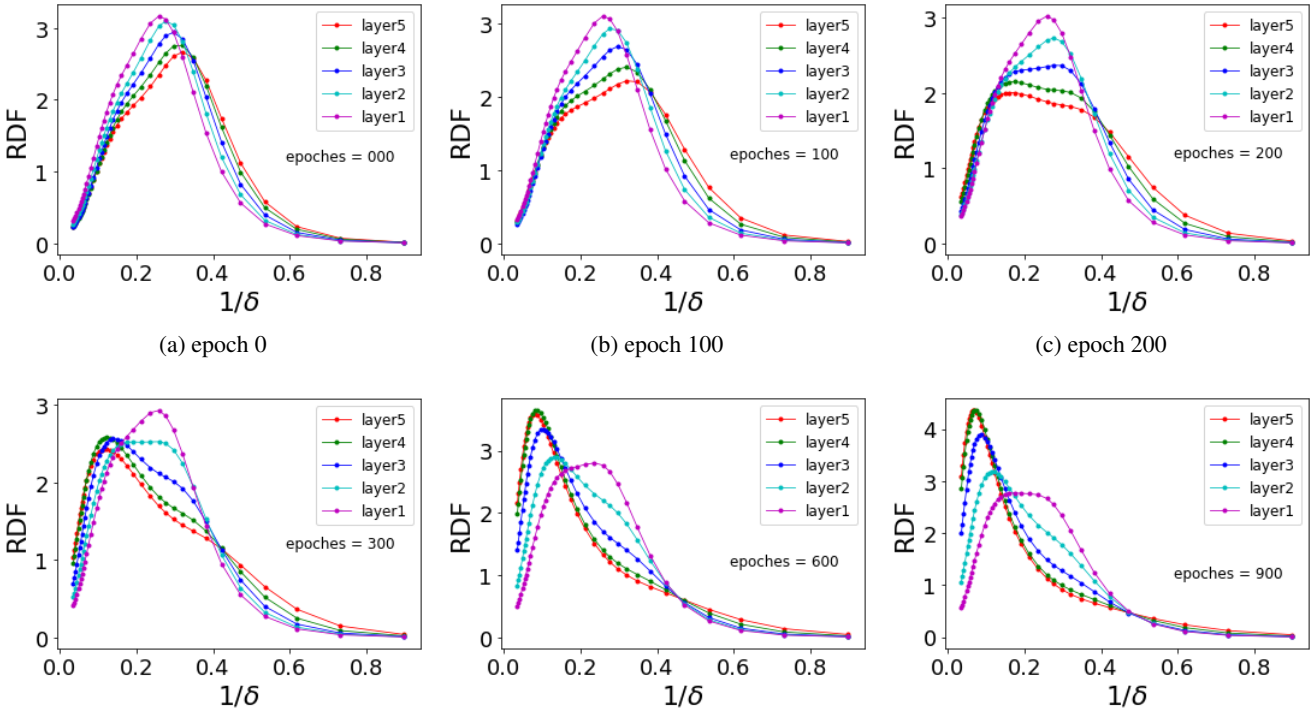
Figure 6: RDF of $S^{[-3]}$ (effective target function of layer "-2") vs. $1/\delta$ at different epochs for variants of Resnet18.



(a) epoch 0 (b) epoch 1 (c) epoch 2

(d) epoch 3 (e) epoch 15 (f) epoch 80

Figure 7: RDF of $\left\{\left(f_{\boldsymbol{\theta}}^{[-3]}(\boldsymbol{x}_i), f_{\boldsymbol{\theta}}^{[-1]}(\boldsymbol{x}_i)\right)\right\}_{i=1}^{n}$ vs. $1/\delta$ at different epochs for variants of Resnet18.



(a) epoch 0 (b) epoch 100 (c) epoch 200

(d) epoch 300 (e) epoch 600 (f) epoch 900

Figure 8: RDF of different hidden layers vs. $1/\delta$ at different epochs for a fully-connected deep neural network when learning MNIST. The five colored curves are for five hidden layers, respectively. The curve with legend "layer i" is the RDF of $S^{[l]}$.

How to characterize the better generalization of deeper network is also a critical problem in deep learning. This work, validating a deep frequency principle, may provide more understanding to this problem in future work. As the network goes deeper, the effective target function for the last hidden layer is more dominated by low frequency. This deep frequency principle phenomenon is widely observed, even in fitting high-frequency function, such as parity function in our experiments. This suggest that deeper network may have more bias towards low frequency. However, it is difficult to examine the frequency distribution of the learned function on the whole Fourier domain due to the high dimensionality of data. In addition, since the generalization increment of a deeper network is more subtle, we are exploring a more precise characterization of the frequency distribution of a high-dimensional function.

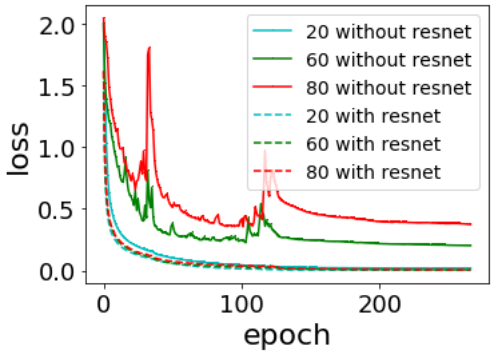
How Deep Is Enough?

The effect of depth can be intuitively understood as a precondition that transforms the target function to a lower frequency function. Qualitatively, it requires more layers to fit a higher frequency function. However, the effect of depth can be saturated. For example, the effective target functions for very deep layers can be very similar in the Fourier domain (dominated by very low frequency components) when the layer number is large enough, as an example shown in Fig. 8(g, h). A too deep network would cause extra waste of computation cost. A further study of the deep frequency principle may also provide a guidance for design the depth of the network structure.

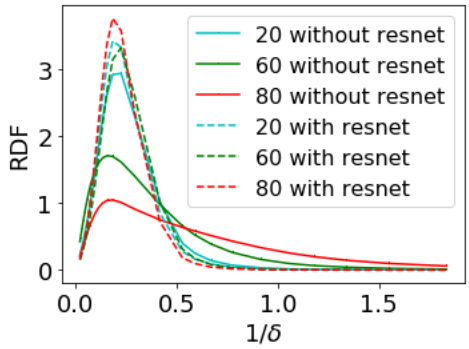
Vanishing Gradient Problem

When the network is too deep, vanishing gradient problem often arises, slows down the training, and deteriorates the generalization (He et al. 2016). As an example, we use very deep fully connected networks, i.e., 20, 60 and 80 layers, to learn MNIST dataset. As shown in Fig. 9(a) (solid lines), deeper networks learn slower in such very deep situations. The frequency distribution of effective target function also violates deep frequency principle in this case. For example, at epoch 267 as shown in Fig. 9(b) (solid lines), the RDFs of $S^{[-3]}$ of different networks show that the effective target function of a deeper hidden layer has more power on high-frequency components. Residual connection is proposed to save deep neural network from vanishing gradient problem and utilize the advantage of depth (He et al. 2016), in which deep frequency principle is satisfied as shown in the previous example in Fig. 6. To verify the effectiveness of residual connection, we add residual connection to the very deep fully connected networks to learn MNIST dataset. As shown in Fig. 9(a) (dashed lines), the learning processes for different networks with residual connections are almost at the same speed. The RDFs in Fig. 9(b) (dashed lines) show that with residual connections, the depth only incurs almost a

negligible effect of deep frequency principle, i.e., a saturation phenomenon. The detailed study of the relation of the depth and the difficulty of the task is left for further research.



(a)



(b)

Figure 9: (a) Training loss vs. epoch. (b) RDF of $S^{[-3]}$ vs. $1/\delta$ at epoch 267. Legend is the number of layers of the fully-connected network with ("with resnet") or without ("without resnet") residual connection trained to fit MNIST.

Taken together, the deep frequency principle proposed in this work may have fruitful implication for future study of deep learning. A detailed study of deep frequency principle may require analyze different dynamical regimes of neural networks. As an example, a recent work (Luo et al. 2020) draws a phase diagram for two-layer ReLU neural networks at infinite width limit by three regimes, linear, critical and condensed regimes. Such study could inspire the study of phase diagram of deep neural networks. The linear regime is well studied (Jacot, Gabriel, and Hongler 2018; Zhang et al. 2019, 2020; Arora et al. 2019), which may be a good starting point and shed lights on the study of other regimes.

Acknowledgements

Z.X. is supported by National Key R&D Program of China (2019YFA0709503), Shanghai Sailing Program, Natural Science Foundation of Shanghai (20ZR1429000), NSFC 62002221 and partially supported by HPC of School of Mathematical Sciences and Student Innovation Center at Shanghai Jiao Tong University.

References

Arora, S.; Cohen, N.; and Hazan, E. 2018. On the optimization of deep networks: Implicit acceleration by overparameterization. In *35th International Conference on Machine Learning*.

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

Arpit, D.; Jastrzebski, S.; Ballas, N.; Krueger, D.; Bengio, E.; Kanwal, M. S.; Maharaj, T.; Fischer, A.; Courville, A.; Bengio, Y.; et al. 2017. A closer look at memorization in deep networks. *International Conference on Machine Learning* .

Biland, S.; Azevedo, V. C.; Kim, B.; and Solenthaler, B. 2019. Frequency-Aware Reconstruction of Fluid Simulations with Generative Networks. *arXiv preprint arXiv:1912.08776* .

Bordelon, B.; Canatar, A.; and Pehlevan, C. 2020. Spectrum Dependent Learning Curves in Kernel Regression and Wide Neural Networks. *arXiv preprint arXiv:2002.02561* .

Cai, W.; Li, X.; and Liu, L. 2019. A phase shift deep neural network for high frequency approximation and wave problems. *Accepted by SISC, arXiv:1909.11759* .

Cao, Y.; Fang, Z.; Wu, Y.; Zhou, D.-X.; and Gu, Q. 2019. Towards Understanding the Spectral Bias of Deep Learning. *arXiv preprint arXiv:1912.01198* .

E, W.; Ma, C.; and Wu, L. 2020. Machine learning from a continuous viewpoint, I. *Science China Mathematics* 1–34.

E, W.; and Qingcan, W. 2018. Exponential convergence of the deep neural network approximationfor analytic functions. *SCIENCE CHINA Mathematics* 61(10): 1733.

Eldan, R.; and Shamir, O. 2016. The power of depth for feedforward neural networks. In *Conference on learning theory*, 907–940.

Gissin, D.; Shalev-Shwartz, S.; and Daniely, A. 2019. The Implicit Bias of Depth: How Incremental Learning Drives Generalization. In *International Conference on Learning Representations*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Jacot, A.; Gabriel, F.; and Hongler, C. 2018. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, 8571–8580.

Jagtap, A. D.; Kawaguchi, K.; and Karniadakis, G. E. 2020. Adaptive activation functions accelerate convergence in deep and physics-informed neural networks. *Journal of Computational Physics* 404: 109136.

Kalimeris, D.; Kaplun, G.; Nakkiran, P.; Edelman, B.; Yang, T.; Barak, B.; and Zhang, H. 2019. SGD on Neural Networks Learns Functions of Increasing Complexity. In *Advances in Neural Information Processing Systems*, 3491–3501.

Kawaguchi, K.; Huang, J.; and Kaelbling, L. P. 2019. Effect of depth and width on local minima in deep learning. *Neural computation* 31(7): 1462–1498.

Li, X.-A.; Xu, Z.-Q. J.; and Zhang, L. 2020. A Multi-Scale DNN Algorithm for Nonlinear Elliptic Equations with Multiple Scales. *Communications in Computational Physics* 28(5): 1886–1906.

Liu, Z.; Cai, W.; and Xu, Z.-Q. J. 2020. Multi-Scale Deep Neural Network (MscaleDNN) for Solving Poisson-Boltzmann Equation in Complex Domains. *Communications in Computational Physics* 28(5): 1970–2001.

Luo, T.; Ma, Z.; Xu, Z.-Q. J.; and Zhang, Y. 2019. Theory of the Frequency Principle for General Deep Neural Networks. *arXiv preprint arXiv:1906.09235* .

Luo, T.; Xu, Z.-Q. J.; Ma, Z.; and Zhang, Y. 2020. Phase diagram for two-layer ReLU neural networks at infinite-width limit. *arXiv preprint arXiv:2007.07497* .

Ma, C.; Wu, L.; and E, W. 2020. The Slow Deterioration of the Generalization Error of the Random Feature Model. In *Proceedings of Machine Learning Research*.

Mingard, C.; Skalse, J.; Valle-Pérez, G.; Martínez-Rubio, D.; Mikulik, V.; and Louis, A. A. 2019. Neural networks are a priori biased towards Boolean functions with low entropy. *arXiv preprint arXiv:1909.11522* .

Rahaman, N.; Arpit, D.; Baratin, A.; Draxler, F.; Lin, M.; Hamprecht, F. A.; Bengio, Y.; and Courville, A. 2019. On the Spectral Bias of Deep Neural Networks. *International Conference on Machine Learning* .

Ronen, B.; Jacobs, D.; Kasten, Y.; and Kritchman, S. 2019. The convergence rate of neural networks for learned functions of different frequencies. In *Advances in Neural Information Processing Systems*, 4763–4772.

Saxe, A. M.; Mcclelland, J. L.; and Ganguli, S. 2014. Exact solutions to the nonlinear dynamics of learning in deep linear neural network. In *In International Conference on Learning Representations*. Citeseer.

Shin, Y. 2019. Effects of Depth, Width, and Initialization: A Convergence Analysis of Layer-wise Training for Deep Linear Neural Networks. *arXiv preprint arXiv:1910.05874* .

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

Valle-Perez, G.; Camargo, C. Q.; and Louis, A. A. 2018. Deep learning generalizes because the parameter-function map is biased towards simple functions. *arXiv preprint arXiv:1805.08522* .

Wang, F.; Eljarrat, A.; Müller, J.; Henninen, T. R.; Erni, R.; and Koch, C. T. 2020. Multi-resolution convolutional neural networks for inverse problems. *Scientific reports* 10(1): 1–11.

Xu, Z. J. 2018. Understanding training and generalization in deep learning by Fourier analysis. *arXiv preprint arXiv:1808.04295* .

Xu, Z.-Q. J.; Zhang, Y.; Luo, T.; Xiao, Y.; and Ma, Z. 2020. Frequency Principle: Fourier Analysis Sheds Light on Deep Neural Networks. *Communications in Computational Physics* 28(5): 1746–1767.

Xu, Z.-Q. J.; Zhang, Y.; and Xiao, Y. 2019. Training behavior of deep neural network in frequency domain. *International Conference on Neural Information Processing* 264–274.

Yang, G.; and Salman, H. 2019. A fine-grained spectral perspective on neural networks. *arXiv preprint arXiv:1907.10599* .

Zhang, Y.; Xu, Z.-Q. J.; Luo, T.; and Ma, Z. 2019. Explicitizing an Implicit Bias of the Frequency Principle in Two-layer Neural Networks. *arXiv preprint arXiv:1905.10264* .

Zhang, Y.; Xu, Z.-Q. J.; Luo, T.; and Ma, Z. 2020. A type of generalization error induced by initialization in deep neural networks. In *Mathematical and Scientific Machine Learning*, 144–164. PMLR.