Rethink the Connections among Generalization, Memorization, and the Spectral Bias of DNNs

Xiao Zhang[1] , **Haoyi Xiong**[2] , **Dongrui Wu**[1]
[1] Huazhong University of Science and Technology, Wuhan, China
[2] Baidu Research, Bejing, China
xiao_zhang@hust.edu.cn, xionghaoyi@baidu.com, drwu@hust.edu.cn

Abstract

Over-parameterized deep neural networks (DNNs) with sufficient capacity to memorize random noise can achieve excellent generalization performance, challenging the bias-variance trade-off in classical learning theory. Recent studies claimed that DNNs first learn simple patterns and then memorize noise; some other works showed a phenomenon that DNNs have a spectral bias to learn target functions from low to high frequencies during training. However, we show that the monotonicity of the learning bias does not always hold: under the experimental setup of deep double descent, the high-frequency components of DNNs diminish in the late stage of training, leading to the second descent of the test error. Besides, we find that the spectrum of DNNs can be applied to indicating the second descent of the test error, even though it is calculated from the training set only.

1 Introduction

The bias-variance trade-off in classical learning theory suggests that models with large capacity to minimize the empirical risk to almost zero usually yield poor generalization performance. However, this is not the case of modern deep neural networks (DNNs): Zhang *et al.* [2017] showed that over-parameterized networks have powerful expressivity to completely memorize all training examples with random labels, yet they can still generalize well on normal examples. This phenomenon cannot be explained by the VC dimension or Rademacher complexity theory.

Some studies attributed this counterintuitive phenomenon to the implicit learning bias of DNN's training procedure: despite of the large hypothesis class of DNNs, stochastic gradient descent (SGD) has an inductive bias to search the hypothesises which show excellent generalization performances. Arpit *et al.* [2017] claimed that DNNs learn patterns first, and then use brute-force memorization to fit the noise hard to generalize. Using mutual information between DNNs and linear models, Kalimeris *et al.* [2019] showed that SGD on DNNs learns functions of increasing complexity gradually. Furthermore, some studies showed that lower frequencies in the input space are learned first and then the higher ones, which is known as the spectral bias or frequency principle of DNNs [Rahaman *et al.*, 2019; Xu *et al.*, 2019a; Xu *et al.*, 2019b]. They showed that overfitting happens when the complexity of models keeps increasing or high-frequency components remain being introduced.

All the findings above are based on a basic assumption that the learning bias in training DNNs is monotonic, e.g., from simple to complex or from low frequencies to high frequencies. However, the monotonicity of the training procedure was recently challenged by epoch-wise double descent: the generalization error first has a classical U-shaped curve and then follows a second descent [Nakkiran *et al.*, 2020]. It is intriguing because according to spectral bias, with higher-frequency components being gradually introduced in training, the generalization performance should deteriorate monotonically due to the memorization of noise.

To better understand the connections among generalization, memorization and the spectral bias of DNNs, we explored the frequency components of learned functions under the experimental setup of double descent (randomly shuffle the labels of part of the training set and train DNNs for an extended number of epochs). We surprisingly observed that at a certain epoch, usually around the start of the second descent, while the perturbed part is still being memorized, the high-frequency components begin to diminish (see the *second descent* phase in Figure 1). After exploring this phenomenon in a traceable toy task, we show that it happens because the prediction surface off the training data manifold becomes flatter and more regularized in the late training stage, which improves the generalization performance of models on the test points that are not covered by the training data manifold. As a result, the second descent of the test error happens.

We further show that though it does not change monotonically, the spectrum manifests itself as an indicator of the training procedure. We trained different DNNs on some image classification datasets, verifying the connections between the second descent of the test error and the diminishment of the high-frequency components, even if the spectrum is calculated from the training set. It suggests that monitoring the test behaviors with only the training set is possible, which provides a novel perspective to studying the generalization and memorization of DNNs in both theory and practice.

The remainder of the paper is organized as follows: we first present our analysis of the spectra of DNNs, then we

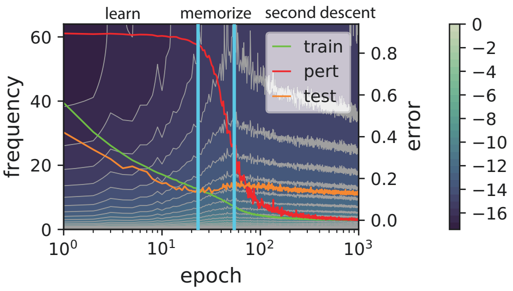


Figure 1: Heat map with contour lines showing the energy ratio (in logarithmic scale) of different frequency components at every epoch, synchronized with training, perturbed, and test errors. The model was ResNet18 trained on CIFAR10 with 10% label noise (Adam with learning-rate 1e-4 for 1,000 epochs). The horizontal axis is shown in logarithmic scale. The vertical lines indicate early stopping and the start of the second descent, dividing the training procedure into three phases.

show that it is possible to monitor the test behaviors according to the spectrum. The following section summarizes some related works. The last section draws conclusions.

Contributions. Our major contributions are highlighted as follows:

- We provide empirical evidence to show that the monotonicity of the learning bias does not always hold. In the late stage of training, though high-frequency components are introduced to memorize the label noise, the prediction surface off the training data manifold is more biased towards low-frequency components, leading to the second descent of the test error.
- We find that unlike errors, the spectra of DNNs vary consistently on the training sets and test sets.
- We explore the test curve and the spectrum calculated on the training set, and discover a correlation between the peak of the spectrum and the start of the second descent of the test error, which suggests that it may be possible to monitor the test behaviors using the training set only.

2 Frequency Components in Training

In this section, we analyze the spectra of DNNs trained on several image classification benchmarks, and find that the monotonicity of the spectral bias does not always hold. Moreover, we show that the diminishment of the high-frequency components leads to the second descent of the test error.

2.1 Fourier Spectrum

Fourier transforming on DNNs can be a tough task due to the high dimensionality of the input space. Rahaman *et al.* [2019] exploited the continuous piecewise-linear structure of ReLU networks to rigorously evaluate the Fourier spectrum, but their approach cannot be performed on DNNs trained on high-dimensional datasets. Xu *et al.* [2019a] used non-uniform discrete Fourier transform (DFT) to capture the global spectrum of a dataset, but it cannot be very accurate due to the sparsity of points in high-dimensional space.

In this paper, we propose a heuristic but more practical metric to measure the spectrum of a DNN. Instead of capturing the frequency components of the whole input space, we pay more attention to the variations of the DNN in local areas around data points. We shall denote the input point sampled from a distribution $\mathcal{D}$ by $\boldsymbol{x} \sim \mathcal{D}$, a normalized random direction by $\boldsymbol{v}_x$, the c-th logit output of a DNN by $f_c(\boldsymbol{x})$, where $c \in \{1, 2, ..., C\}$ and C is the number of classes. We evenly sample N points from $[\boldsymbol{x} - h\boldsymbol{v}_x, \boldsymbol{x} + h\boldsymbol{v}_x]$ to perform the discrete Fourier transform, where h bounds the area. The Fourier transform of $f_c(\boldsymbol{x})$ is then:

$$\tilde{f}_{c,\boldsymbol{x}}(k) = \sum_{n=1}^{N} f_c\left(\boldsymbol{x} + \frac{2n - N - 1}{N - 1} h\boldsymbol{v}_x\right) e^{-i2\pi \frac{n}{N} k}. \quad (1)$$

We use the logit outputs instead of the probabilities so that the spectrum is irrelevant to the rescaling of the DNN parameters. Because, when the weights of the last layer are multiplied by $\alpha > 0$ (this operation does not change the decision boundary), the spectrum will change nonlinearly if we use the probability outputs. We then add up the spectra across the dataset and the logit outputs to illustrate the local variation from a global viewpoint:

$$A_k = \frac{1}{C} \sum_{c=1}^{C} \mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}} \left| \tilde{f}_{c,\boldsymbol{x}}(k) \right|^2. \quad (2)$$

In practice, we only need a small number (as tested in our experiments, 500 data points are usually enough) of data points to approximate the expectation of $\left| \tilde{f}_{c,\boldsymbol{x}}(k) \right|^2$ in (2).

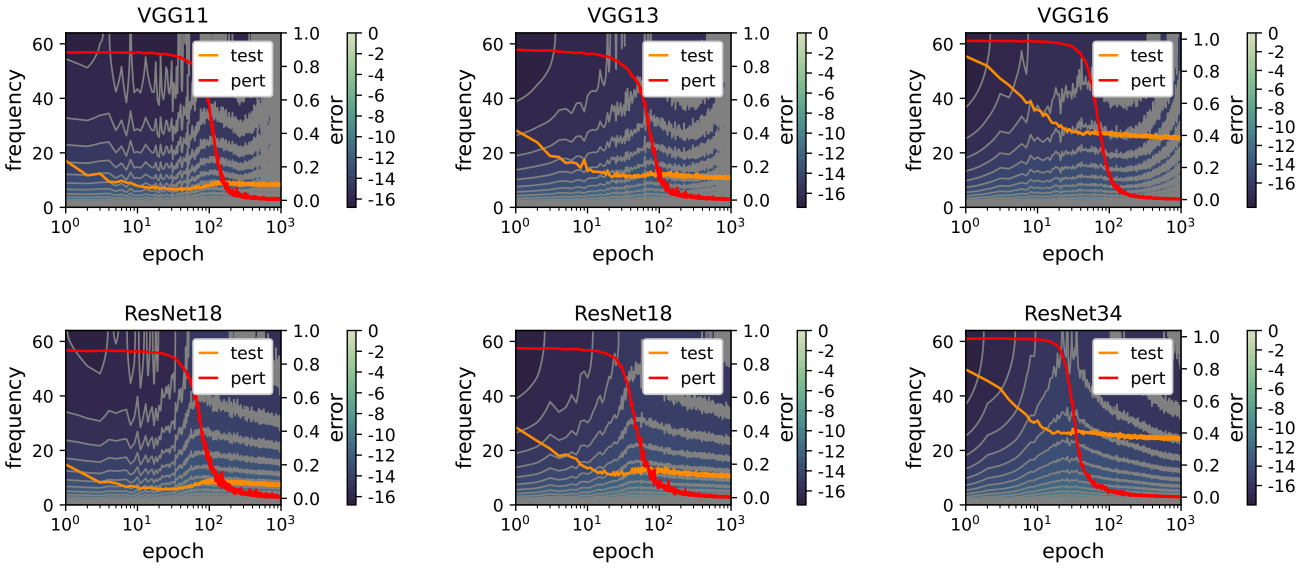
Note that computing A_k does not need any label information of the dataset, suggesting that A_k may be applied to semi-supervised or unsupervised learning.

2.2 Non-Monotonicity of the Spectral Bias

Previous studies on spectral bias assumed that its evolutionary process is monotonic, i.e., DNNs learn the low frequencies first, and then the high frequencies, which means we should observe a monotonic increase in the ratio of high-frequency components. However, this is not what we observed: we found that at a certain epoch, usually around the epoch when the second descent starts, the ratio of high-frequency components begins to diminish.

Our experimental setup was analogous to deep double descent [Nakkiran *et al.*, 2020]. We here consider two architectures[1] (VGG [Simonyan and Zisserman, 2015] and ResNet [He *et al.*, 2016]) on three image datasets (SVHN [Netzer *et al.*, 2011], CIFAR10 and CIFAR100 [Krizhevsky, 2009]). We normalized the values of image pixels to $[0, 1]$ and randomly shuffled 10% labels of the training set to strengthen double descent, where the perturbed part was denoted by *perturbed set*. The batch-size was set to 128, and we utilized the Adam optimizer with the learning rate 0.0001 to train the models for 1,000 epochs with data augmentation.

[1] Adapted from https://github.com/kuangliu/pytorch-cifar.



(a) SVHN

(b) CIFAR10

(c) CIFAR100

Figure 2: ResNet and VGG trained on SVHN, CIFAR10 and CIFAR100 with 10% label noise. The red curves indicate the errors on the perturbed set, whereas the orange ones indicate the errors on the test set. The heat map with contour lines depicts the spectrum R_k, which is calculated on the training set. The horizontal axis is shown in logarithmic scale. Appendix A presents results of other levels of label noise.

To better compare the spectrum along the training process, we calculated the energy ratio of A_k in logarithmic scale:

$$R_k = \log \frac{A_k}{\sum_j A_j}. \tag{3}$$

For every epoch, we randomly chose 500 data points from the training set, sampled a normalized direction $\boldsymbol{v}_x$ for each data point, and calculated R_k for the k-th frequency component[2] based on the sampled data points and the directions. In our experiments, we set $h = 0.5$ and $N = 128$.

The spectra and learning curves are shown in Figure 2. From the error curves, we can observe that the test error decreases very quickly at the beginning of the training, whereas the error on the perturbed set remains high, suggesting that the models effectively learn the patterns of the data in this period. However, as the training goes on, the error on the perturbed set decreases rapidly, along with a little increase of the test error. In this period, the models start to memorize the noise, which leads to overfitting on the training set. So far, the behaviors of the models are consistent with the conventional wisdom. However, if the models are trained with more epochs, the peak of the test error occurs, just around the epoch when the model memorizes the noise, and then the test error steps into the second descent. This is known as the epoch-wise double descent [Nakkiran *et al.*, 2020].

The heat map of R_k in Figure 2 presents a new perspective to looking at this phenomenon. The spectral bias manifests itself in the learning and memorization phases: the models first introduce low-frequency components and then the high-frequency ones. The ratio of the high-frequency components increases rapidly when the models try to memorize the noise. Nonetheless, along the second descent of the test error, the high-frequency components begin to diminish, violating the claims about the monotonicity of the spectral bias. The non-monotonicity of the spectral bias implies that limited high-frequency components may be sufficient to memorize the noise, which can be observed during the second descent: though the ratio of the high-frequency components decreases, the perturbed set is still being memorized.

Our experimental results also show that the skip connection significantly influences the spectra. Comparing the late-stage heat maps between VGG and ResNet in Figure 2, we can observe that ResNet seems to have a bias towards low-frequency components. Since high-frequency components usually suggest complex decision boundary with poor generalization, this may explain why skip connection can improve the performance of DNNs.

2.3 Why Do High-Frequency Components Diminish?

Label noise raises a potential demand for the high-frequency components. It is reasonable since the perturbed point resembles the Dirac Delta function on the prediction surface of the training data manifold, which has a broadband power spectrum. However, this explanation may lead to a new question: *why do high-frequency components diminish when the perturbed set is still memorized*?

Notice that the perturbed set remains memorized during the second descent, which means the gain of the generalization performance in this phase is not from fitting the training

[2] We call k “frequency” in the rest of the paper.

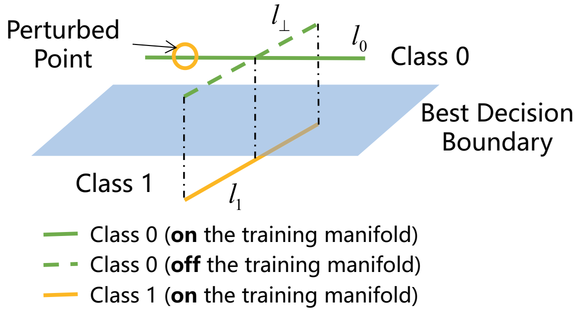

Perturbed Point
$l_\perp$
l_0
Class 0
Best Decision Boundary
Class 1
l_1
Class 0 (**on** the training manifold)
Class 0 (**off** the training manifold)
Class 1 (**on** the training manifold)

Figure 3: Illustration of the toy task. The blue plane is the desired decision boundary separating l_0 and l_1. The perturbed point is labeled as Class 1. The off-manifold line $l_\perp$ intersects with the on-manifold line l_0 perpendicularly. The task is to train the model on l_0 and l_1, and observe the spectra on l_0 and $l_\perp$.

data manifold but from regularizing the prediction surface in other input space. Inspired by this fact, we consider the spectra along the training data manifold and off the training data manifold separately. We show that the complex variation of the spectrum shown in Section 2.2 is the combination of two processes: the on-manifold prediction surface keeps introducing high-frequency components to fit the perturbed points, whereas the off-manifold prediction surface gradually becomes biased towards low-frequency components.

To verify our hypothesis, we design a toy task which requires separating two disjoint but perpendicular lines that lie in the three-dimensional space (the label of one data point is perturbed), and then we investigate the variation of the on-/off- manifold spectra and generalization performances. Figure 4 presents a simple illustration of our toy task.

Specifically, we define l_0 and l_1 as:

$$l_i = \{\boldsymbol{r}_i + k \cdot \boldsymbol{v}_i | -1 \le k \le 1\}, i \in \{0, 1\}, \tag{4}$$

where $\boldsymbol{r}_0 = [0.1, 0.1, 0.1]$, $\boldsymbol{r}_1 = -\boldsymbol{r}_0$, $\boldsymbol{v}_0 = \frac{1}{\sqrt{2}}[1, -1, 0]$, and $\boldsymbol{v}_1 = \frac{1}{\sqrt{6}}[1, 1, -2]$. For the training set, we evenly sample 51 points from l_0 and l_1, respectively, and label the points in l_i as Class i (except one randomly chosen perturbed point in l_0). For the test set, we evenly sample 201 points from l_0 and l_1, respectively, without any perturbation. A ReLU DNN with two fully-connected hidden layers (100 neurons for each) is applied to train on the training set with Adam optimizer (lr=5e-4). During the training process, we compute both the on-manifold spectrum on l_0 and the off-manifold spectrum calculated on $l_\perp = \{\boldsymbol{r}_0 + k \cdot \boldsymbol{v}_1 | -1 \le k \le 1\}$. Note that here we can directly perform DFT on the corresponding line to obtain the precise spectrum.

Figure 4 presents the on-/off- manifold spectra and the generalization accuracy. As shown in Figure 4(a), the on-manifold spectrum keeps introducing high-frequency components to memorize the perturbed point, which is not the case for the off-manifold spectrum. Figure 4(b) shows that after the perturbed point is memorized, the off-manifold spectrum is more biased towards the low-frequency components, resulting in a much flatter off-manifold prediction surface. The regularized prediction surface has low complexity and potentially improves the model's generalization performance on some test points which are not covered by the training data manifold, leading to the second descent of the test error. As illustrated in Figure 4(c), the on-manifold accuracy slightly decreases after memorizing the perturbed point, whereas the off-manifold accuracy keeps increasing in the same period.

3 Monitor the Test Curve without Test Set

In Figure 2, we have seen that the peak of R_k seems to synchronize with the start of the second descent of the test error. Based on this observation, we show that it may be possible to monitor the second descent of test error without any test set.

3.1 Consistency of R_k on Training and Test Sets

One obstacle of using training errors to monitor test ones is that they do not vary consistently during training, especially when the number of epochs is large. This inconsistency is a result of the direct involvement of training errors in the optimization function, whereas test errors are excluded. Therefore, if we want to use a metric calculated on the training set to monitor the behaviors of the test curve, the most essential step is to cut off its connection to the optimization function.

Apparently, R_k, which measures the local variation of DNNs, satisfies this requirement. More importantly, R_k is calculated without any label information, which further weakens its links to the optimization function. To verify its consistency on the training and test sets, we calculated R_k on the test set, which is presented in Appendix B and shows no significant difference with R_k calculated on the training set (see Figure 2). To compare the consistency of training and testing curves for errors and R_k, we calculated their short-time Pearson correlation coefficients (PCCs), whose value at training epoch t is the PCC in a sliding window of length l (set to 100 in our experiments) at t:

$$p_l(t; \boldsymbol{x}, \boldsymbol{y}) = \frac{\sum_{i=t}^{t+l-1}(\boldsymbol{x}_i - \bar{\boldsymbol{x}}_t)(\boldsymbol{y}_i - \bar{\boldsymbol{y}}_t)}{\sqrt{\sum_{i=t}^{t+l-1}(\boldsymbol{x}_i - \bar{\boldsymbol{x}}_t)^2 \sum_{i=t}^{t+l-1}(\boldsymbol{y}_i - \bar{\boldsymbol{y}}_t)^2}}, \tag{5}$$

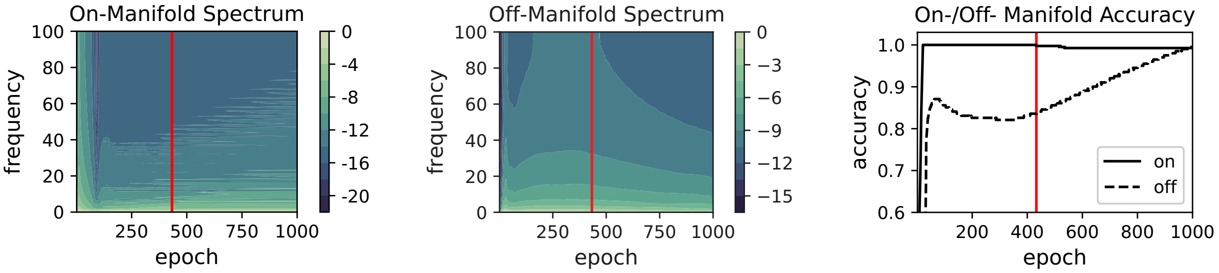
where $\bar{\boldsymbol{x}}_t = \frac{1}{l}\sum_{j=t}^{t+l-1} \boldsymbol{x}_j$ and $\bar{\boldsymbol{y}}_t = \frac{1}{l}\sum_{j=t}^{t+l-1} \boldsymbol{y}_j$.

Figure 5 shows short-time PCCs of the training and test errors, and R_k on the training and test sets as well. Compared with the error, R_k shows better consistency on the training and testing sets. Observe that the short-time PCC decreases when the number of training epochs is very large, because in the late stage, the variation tendency of the errors or R_k is so slow that the noise of variation dominates the short-time PCCs instead of the overall variation tendency.

From the above observation and the discussions in the last section, we can conclude that the local variation of DNNs seems to have a consistent pattern in the input space during training, which makes R_k an excellent metric to indicate the training progress.

3.2 Use R_k to Monitor the Second Descent of the Test Error

We have verified the consistency of R_k on training and test sets. To monitor the test curves, the metric should also have



(a) (b) (c)

Figure 4: On-/off- manifold spectra and accuracies. The red lines indicate the epoch when the perturbed point begins to be memorized.

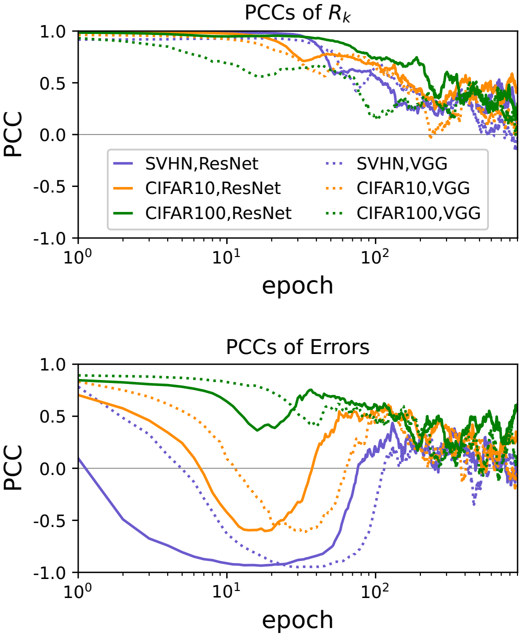


Figure 5: Short-time PCCs of R_k and errors. The colors represent different datasets, whereas the styles of lines indicate different categories of architectures. The short-time PCC of R_k is averaged over $k = 1, 2..., 64$. The horizontal axis is shown in logarithmic scale.

nontrivial meaningful connections to the test behaviors of DNNs. In Figure 2, we can clearly observe the synchronization of the peak of R_k, which is calculated on the training set, with the start of the second descent of the test error. This subsection further explores this connection.

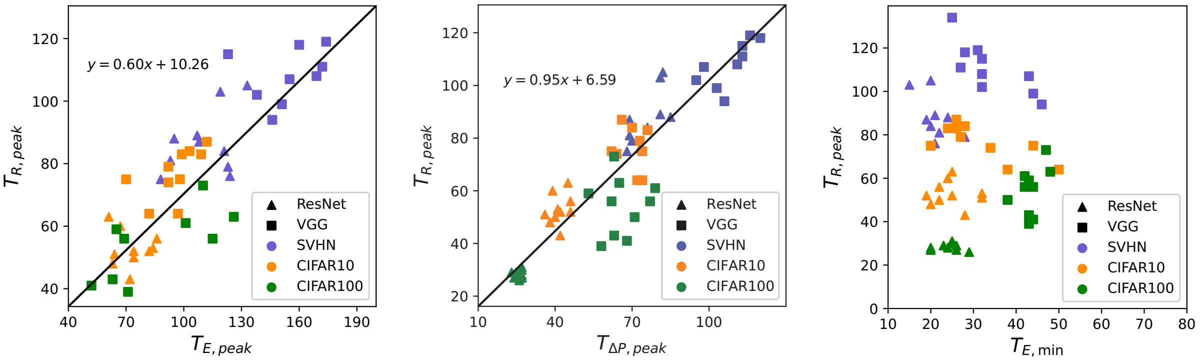
Let $R_{k,t}$ denote R_k at the t-th epoch ($R_{k,t}$ was smoothed by mean value filtering with a window length of 10 epochs), and E_t the corresponding test error. Similar to early stopping, we searched the peak of E_t and $R_t = \sum_k \alpha_k R_{k,t}$ with the patience of 30 epochs, where α_k is the weight of $R_{k,t}$ and all set to 1 in our experiments. More specifically, we performed early stopping twice: first, we tried to find the number of epochs for minimal R_t and E_t, denoted by $T_{R,\min}$ and $T_{E,\min}$, respectively; then, we used $T_{R,\min}$ and $T_{E,\min}$ as start points to search for the epochs of maximal R_t and E_t, denoted by $T_{R,\text{peak}}$ and $T_{E,\text{peak}}$, respectively.

We trained different models on several datasets (SVHN: ResNet18 and VGG11; CIFAR10: ResNet18 and VGG13; CIFAR100: ResNet34 and VGG16) with different levels of label noise (10% and 20%) for five runs, and explored the relationship between the spectra and the test behaviors. As shown in Figure 6(a), despite of different models and datasets, we can clearly observe a linear positive correlation between $T_{R,\text{peak}}$ and $T_{E,\text{peak}}$, suggesting that it is possible to predict the second descent of the test error with only the training set. Moreover, the spectrum can also indicate how the peak of the double descent moves when the model width varies, which further verifies the connection between the spectrum and the epoch-wise double descent (see Appendix C for more details).

$T_{R,\text{peak}}$ is also related to the decreasing rate of errors on the perturbed set. Let P_t denote the perturbed error at the t-th epoch, and $\Delta P_t = -(P_t - P_{t-1})$ the decreasing rate. We searched the peak of $\overline{\Delta P}_t = \frac{1}{2\Delta T+1} \sum_{i=-\Delta T}^{\Delta T} \Delta P_{t+i}$ ($\Delta T = 5$ in our experiments) due to the large variance of ΔP_t, and the corresponding epoch was denoted by $T_{\Delta P,\text{peak}}$. Figure 6(b) shows that when high-frequency components reach their largest ratio, the perturbed error decreases the fastest, suggesting that the spectrum can also be applied to studying some more subtle behaviors.

3.3 Discussion

Overfitting is another important test behavior. A validation set is usually reserved to indicate when overfitting happens. However, this may reduce the training performance because of the reduced training set size. A more popular approach is that after obtaining a suitable number of training epochs from the validation set, we combine the training set and the validation set to train the model for the same number of epochs. However, this approach also has two shortcomings: 1) there is no guarantee that the "sweet spot" of the training epoch does not change when the training and validation sets are combined; and, 2) it is time-consuming to train the model several times. If we can discover a novel metric which can be calculated on the training set but effective enough to predict the epoch of overfitting (just like the spectrum to the second descent of the test error), these problems can be easily solved.



(a)

(b)

(c)

Figure 6: (a) $T_{R,\text{peak}}$ w.r.t. $T_{E,\text{peak}}$; (b) $T_{R,\text{peak}}$ w.r.t. $T_{\Delta P,\text{peak}}$; (c) $T_{R,\text{peak}}$ w.r.t. $T_{E,\min}$. The colors represent different datasets, whereas the shapes indicate different categories of DNNs, e.g., green squares represent VGG on CIFAR100. The black line is fitted by linear regression. Each experiment was repeated five times. The Pearson correlation coefficients for (a) and (b) are 0.88 and 0.92, respectively.

However, we are not able to find a direct connection between $T_{R,\text{peak}}$ and $T_{E,\min}$ [see Figure 6(c)]. It is also one of our future research directions to find a metric to indicate $T_{E,\min}$.

4 Related Work

There are many studies related to the themes of this paper:

Generalization and Memorization. Over-parameterized DNNs are believed to have large expressivity, usually measured by the number of linear regions in the input space [Pascanu *et al.*, 2014; Montufar *et al.*, 2014; Poole *et al.*, 2016; Arora *et al.*, 2018; Zhang and Wu, 2020]. However, it cannot explain why a DNN, whose capacity is large enough to fit random noise, still has low variance on normal datasets [Zhang *et al.*, 2017]. Arpit *et al.* [2017] examined the role of memorization in deep learning and showed its connections to the model capacity and generalization. Zhang *et al.* [2020] studied the interplay between memorization and generalization and empirically showed that different architectures exhibit different inductive biases. Despite of these studies, memorization and generalization of DNNs is still an open problem requiring more exploration.

Learning Bias. Learning bias suggests that SGD has an implicit inductive bias when searching for the solutions, e.g., from learning patterns to memorizing noise [Arpit *et al.*, 2017], from simple to complex [Kalimeris *et al.*, 2019], or from low frequency to high frequency [Rahaman *et al.*, 2019; Xu *et al.*, 2019a; Xu *et al.*, 2019b]. For spectral bias, some studies investigated the convergence rate of different frequency components, but they were very specific, e.g., using DNNs with infinity width or synthetic datasets [Ronen *et al.*, 2019; Cao *et al.*, 2019]. Moreover, they all suggested that learning bias is monotonic, which is different from our findings.

Double Descent. There is epoch-wise double descent and model-wise double descent [Nakkiran *et al.*, 2020]. The former is a general phenomenon observed by many studies [Belkin *et al.*, 2018; Geiger *et al.*, 2020; Yang *et al.*, 2020], whereas the latter was proposed recently, inspired by a unified notion of "Effective Model Complexity" [Nakkiran *et al.*, 2020]. Our work provides a novel perspective to analyze the epoch-wise double descent.

5 Conclusions

Our research suggests that we need to rethink the connections among generalization, memorization and the spectral bias of DNNs. We studied the frequency components of DNNs in the data point neighbors via Fourier analysis. We showed that the monotonicity of the spectral bias does not always hold, because the off-manifold prediction surface may reduce its high-frequency components in the late training stage. Though perturbed points on the training data manifold remain memorized by the on-manifold prediction surface, this implicit regularization on the off-manifold prediction surface can still help improve the generalization performance. We further illustrated that unlike errors, the spectrum shows remarkable consistency on the training sets and test sets. Based on these observation, we found the potential correlation of the spectrum, calculated on the training set, to the second descent of the test error, suggesting that it may be possible to monitor the test behavior using the training set only.

Our future research will: a) analyze the spectrum of DNNs in other learning tasks, e.g., natural language processing, speech recognition and so on; b) find a new metric which can be easily calculated on the training set, but effective enough to indicate the start of overfitting; c) explore the role that SGD and skip connection play in the spectra of DNNs.

Acknowledgements

This work was supported by the CCF-BAIDU Open Fund under Grant OF2020006, the Technology Innovation Project of Hubei Province of China under Grant 2019AEA171, the National Natural Science Foundation of China under Grants 61873321 and U1913207, and the International Science and Technology Cooperation Program of China under Grant 2017YFE0128300.

References

[Arora *et al.*, 2018] Raman Arora, Amitabh Basu, Poorya Mianjy, and Anirbit Mukherjee. Understanding deep neural networks with rectified linear units. In *Proc. Int'l Conf. on Learning Representations*, Vancouver, Canada, May 2018.

[Arpit *et al.*, 2017] Devansh Arpit, Stanisław Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. A closer look at memorization in deep networks. In *Proc. 34th Int'l Conf. on Machine Learning*, volume 70, pages 233–242, Sydney, Australia, August 2017.

[Belkin *et al.*, 2018] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine learning and the bias-variance trade-off. *CoRR*, abs/1812.11118, 2018.

[Cao *et al.*, 2019] Yuan Cao, Zhiying Fang, Yue Wu, Ding-Xuan Zhou, and Quanquan Gu. Towards understanding the spectral bias of deep learning. *CoRR*, abs/1912.01198, 2019.

[Geiger *et al.*, 2020] Mario Geiger, Arthur Jacot, Stefano Spigler, Franck Gabriel, Levent Sagun, Stéphane d'Ascoli, Giulio Biroli, Clément Hongler, and Matthieu Wyart. Scaling description of generalization with number of parameters in deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2020(2):023401, 2020.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 770–778, Las Vegas, NV, June 2016.

[Kalimeris *et al.*, 2019] Dimitris Kalimeris, Gal Kaplun, Preetum Nakkiran, Benjamin Edelman, Tristan Yang, Boaz Barak, and Haofeng Zhang. SGD on neural networks learns functions of increasing complexity. In *Proc. Advances in Neural Information Processing Systems*, pages 3491–3501, Vancouver, Canada, December 2019.

[Krizhevsky, 2009] Alex Krizhevsky. Learning multiple layers of features from tiny images. techreport, Canadian Institute for Advanced Research, April 2009.

[Montufar *et al.*, 2014] Guido F Montufar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. In *Proc. Advances in Neural Information Processing Systems*, pages 2924–2932, Montreal, Canada, December 2014.

[Nakkiran *et al.*, 2020] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. In *Proc. Int'l Conf. on Learning Representations*, Addis Ababa, Ethiopia, April 2020.

[Netzer *et al.*, 2011] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, Granada, Spain, December 2011.

[Pascanu *et al.*, 2014] Razvan Pascanu, Guido Montufar, and Yoshua Bengio. On the number of response regions of deep feed forward networks with piece-wise linear activations. *CoRR*, abs/1312.6098, 2014.

[Poole *et al.*, 2016] Ben Poole, Subhaneil Lahiri, Maithra Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. In *Proc. Advances in Neural Information Processing Systems*, pages 3360–3368, Barcelona, Spain, December 2016.

[Rahaman *et al.*, 2019] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *Proc. 36th Int'l Conf. on Machine Learning*, pages 5301–5310, Long Beach, CA, May 2019.

[Ronen *et al.*, 2019] Basri Ronen, David Jacobs, Yoni Kasten, and Shira Kritchman. The convergence rate of neural networks for learned functions of different frequencies. In *Proc. Advances in Neural Information Processing Systems*, pages 4763–4772, Vancouver, Canada, December 2019.

[Simonyan and Zisserman, 2015] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proc. Int'l Conf. on Learning Representations*, San Diego, CA, May 2015.

[Xu *et al.*, 2019a] Zhi-Qin John Xu, Yaoyu Zhang, Tao Luo, Yanyang Xiao, and Zheng Ma. Frequency principle: Fourier analysis sheds light on deep neural networks. *CoRR*, abs/1901.06523, 2019.

[Xu *et al.*, 2019b] Zhi-Qin John Xu, Yaoyu Zhang, and Yanyang Xiao. Training behavior of deep neural network in frequency domain. In *Proc. Int'l Conf. on Neural Information Processing*, pages 264–274, Sydney, Australia, December 2019.

[Yang *et al.*, 2020] Zitong Yang, Yaodong Yu, Chong You, Jacob Steinhardt, and Yi Ma. Rethinking bias-variance trade-off for generalization of neural networks. *CoRR*, abs/2002.11328, 2020.

[Zhang and Wu, 2020] Xiao Zhang and Dongrui Wu. Empirical studies on the properties of linear regions in deep neural networks. In *Proc. Int'l Conf. on Learning Representations*, Addis Ababa, Ethiopia, April 2020.

[Zhang *et al.*, 2017] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *Proc. Int'l Conf. on Learning Representations*, Toulon, France, April 2017.

[Zhang *et al.*, 2020] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Michael C Mozer, and Yoram Singer. Identity crisis: Memorization and generalization under extreme overparameterization. In *Proc. Int'l Conf. on Learning Representations*, Addis Ababa, Ethiopia, April 2020.