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

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

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. These suggest some connections among generalization, memorization, and the spectral bias of DNNs: the low-frequency components in the input space represent the *patterns* which can generalize, whereas the high-frequency components represent the *noise* which needs to be memorized. However, we show that this connection does not always hold: under the experimental setup of deep double descent, the high-frequency components of DNNs begin to diminish in the second descent, whereas the examples with random labels are still being memorized. Moreover, we find that the spectrum of DNNs can be applied to monitoring the test behavior, e.g., it can indicate when the second descent of the test error starts, even though the spectrum is calculated from the training set only.

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 (Vapnik 1999). 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. Some other works showed that increasing the depth/width of DNNs can enhance the generalization/optimization, which also conflicts the VC dimension or Rademacher complexity theory (Krizhevsky, Sutskever, and Hinton 2012; Simonyan and Zisserman 2015; He et al. 2016; Zagoruyko and Komodakis 2016; Li et al. 2018).

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. 2019; Xu, Zhang, and Xiao 2019). Through these findings, one may naturally link the low-frequency components to patterns which can generalize and the high-frequency ones to noise which has to be memorized. In this paper, we show that this dose **not** always hold.

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-wised 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 somehow begin to diminish (see the *second descent* phase in Figure 1). In other words, low-frequency components may be sufficient to memorize the noise.

We further show that though it dose 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 mon-

*Dongrui Wu is the corresponding author.

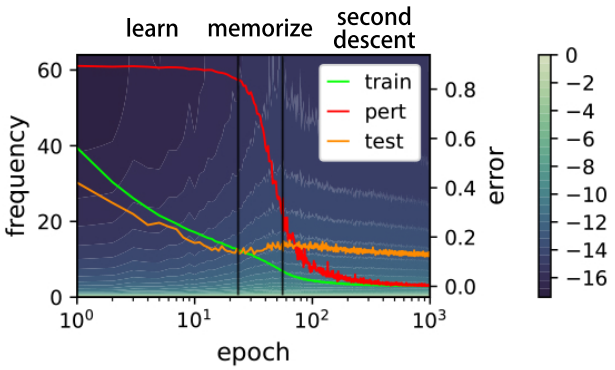


Figure 1: Heat map with contour lines showing the energy ratio (in logarithmic scale) of different frequency components at every epoch, synchronized with training, perturbed, and test errors. The model was ResNet18 trained on CIFAR10 with 10% label noise (Adam (Kingma and Ba 2014) with learning-rate 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.

itoring 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 spectrums of DNNs, then we shows that it is possible to monitor the test behaviors from the training process 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 evidences to show that the monotonicity of the learning bias does not always hold. Based on this, we further show that the low-frequency components are sufficient to memorize data with random labels.
- We find that unlike errors, the spectrums of DNNs vary consistently on the perturbed sets, 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.

Frequency Components in Training

In this section, we analyze the spectrums of different models trained on several image classification benchmarks. We empirically find that the monotonicity of the spectral bias does not always hold and the low-frequency components can also memorize the noise.

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. (2019) used non-uniform discrete Fourier transform to capture the global spectrum of a dataset, but the obtained spectrum may not be accurate due to the sparsity of the data 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. Mathematically, 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 (DFT), 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 spectrums 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).

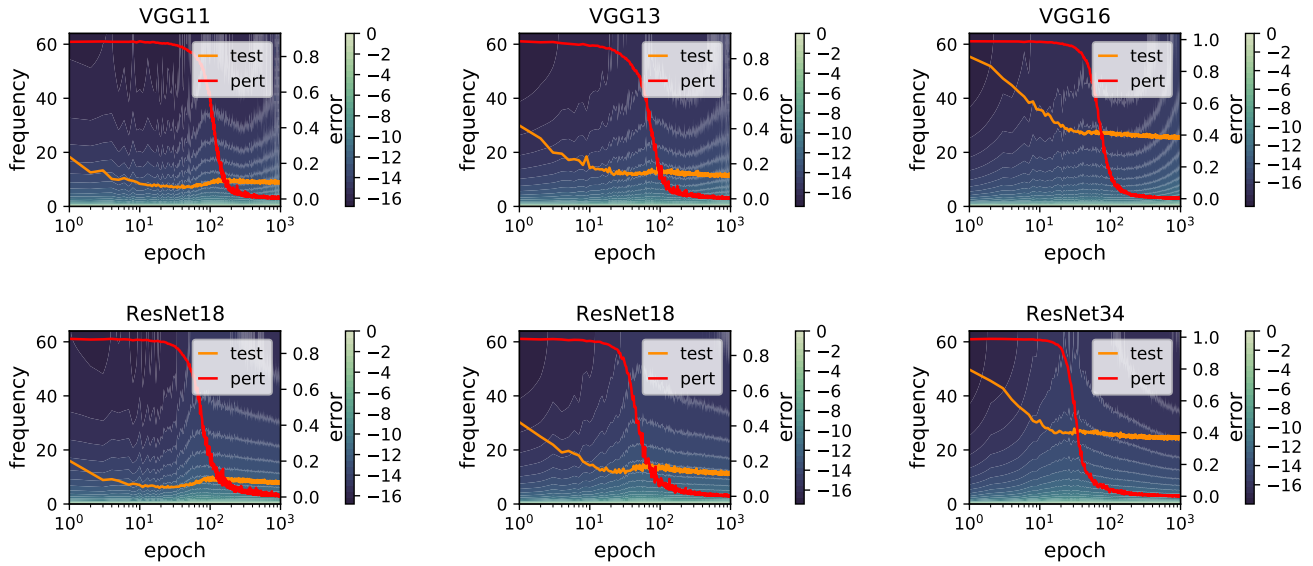
Note that computing A_k does not need any label information of the dataset, suggesting that A_k can be calculated on unlabeled data, i.e., in semi-supervised or unsupervised learning.

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

[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 R_k, which is defined in (3) and calculated on the training set. The horizontal axis is shown in logarithmic scale.

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 (Kingma and Ba 2014) with the learning rate 0.0001 to train the models for $1,000$ epochs with data augmentation.

In order 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 $N = 128$ in (1). The influence of h will be discussed in the next subsection.

The spectrums 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 look 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 the relatively-low-frequency components are 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. Though the models on SVHN behave a little differently at the early epochs, they still fall into above patterns after about ten training epochs.

Discussions

In this subsection, we discuss several factors that influence the spectrum.

Global vs. Local The value of h limits the range of areas where spectrums are expected to be calculated. Considering that the landscape of DNN's outputs on the data manifold is

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

significantly different from that of the data manifold (Zhang and Wu 2020), the ideal way to analyze outputs' variation is to calculate frequency components along the data manifold, such as its geodesic line. However, precisely capturing the data manifold is not an easy task, making this approach less practical. As a compromise, we performed DFT on small local areas around data points, where the properties of Euclidean spaces may remain (according to our statistics, the expectation of the distance from one data point to its closest point in l_2 norm is around 9.0 for CIFAR10 and CIFAR100, and 5.0 for SVHN. Compared with them, $h = 0.5$ is relatively small). As shown in Figure 3, the variation of the spectrum becomes very unstable when h is set to a large value, because most of the areas are off the data manifold.

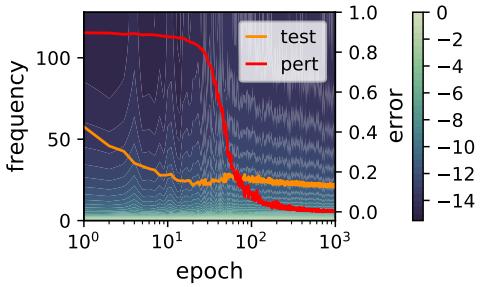


Figure 3: ResNet18 trained on CIFAR10 with 10% label noise ($h = 9$). The red curves indicate the errors on the perturbed set, whereas the orange ones indicate the errors on the test set. The heat map with contour lines depicts R_k, which is calculated on the training set. The horizontal axis is shown in logarithmic scale.

Label Noise Label noise is a fundamental phenomenon naturally existing in a wide range of different forms (Northcutt, Jiang, and Chuang 2019), and our experimental setup amplifies this effect. Label noise increases the complexity of the data manifolds, resulting in a potential demand for the high-frequency components. It is reasonable since the perturbed point resembles the Dirac Delta function in the manifold, which has a broadband power spectrum. To specify the influence of label noise, we trained models with different levels of label noise under the same training setup, and compared their spectrums. Figure 4 shows an example of ResNet18 trained on CIFAR10. As the contour lines indicate, label noise leads to a significant peak of R_k of high-frequency components. Without the label noise, the contour lines are much flatter, suggesting that label noise is an important factor of the high-frequency components. Appendix A presents other spectrums of models trained with different levels of label noise.

The explanation that label noise requires high-frequency components may lead to a new question that why high-frequency components diminish when the second descent starts? One may argue that the spectrum is obtained from the training set, which only contains a small proportion of the perturbed data, and hence though the perturbed points

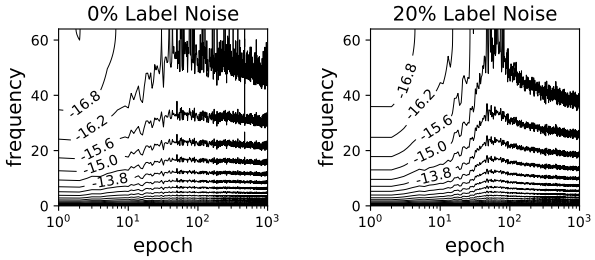


Figure 4: ResNet18 trained on CIFAR10 with 0% and 20% label noise. The numbers indicate the values of contour lines of R_k, which are calculated on the training set. The horizontal axis is shown in logarithmic scale.

still keep the high-frequency components in their neighborhood, the averaged spectrum cannot reflect this property. In fact, this is **not** true. It seems that this phenomenon exists in a global manner. Appendix B shows the spectrum calculated on the perturbed set. We can still observe the diminishment of the high-frequency components. We believe one possible cause is the sparsity of the data in the high-dimensional space. Since the perturbed points lie far away from other sampled points due to the sparsity, the prediction surface is relatively flat in their neighborhood.

Skip Connection It is well-known that the architecture of DNNs introduces strong priors to training and results in some inductive biases. Our experimental results show that the skip connection significantly influence the spectrums. 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. Our finding may inspire the design of new architectures or regularization to improve DNN's generalization performance by repressing high-frequency components.

Monitor the Test Curve without Any Test Set

In Figure 2, we have already 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 test curve without any test set.

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 C 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), which is the PCC of series in a sliding window of length l (set to 100 in our experiments) at the training epoch t:

$$p_l(t;\boldsymbol{x},\boldsymbol{y}) = \frac{\sum_{i=t}^{t+l-1}(\boldsymbol{x}_i-\bar{\boldsymbol{x}}_t)(\boldsymbol{y}_i-\bar{\boldsymbol{y}}_t)}{\sqrt{\sum_{i=t}^{t+l-1}(\boldsymbol{x}_i-\bar{\boldsymbol{x}}_t)^2\sum_{i=t}^{t+l-1}(\boldsymbol{y}_i-\bar{\boldsymbol{y}}_t)^2}}, \tag{4}$$

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. We can see clearly that 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.

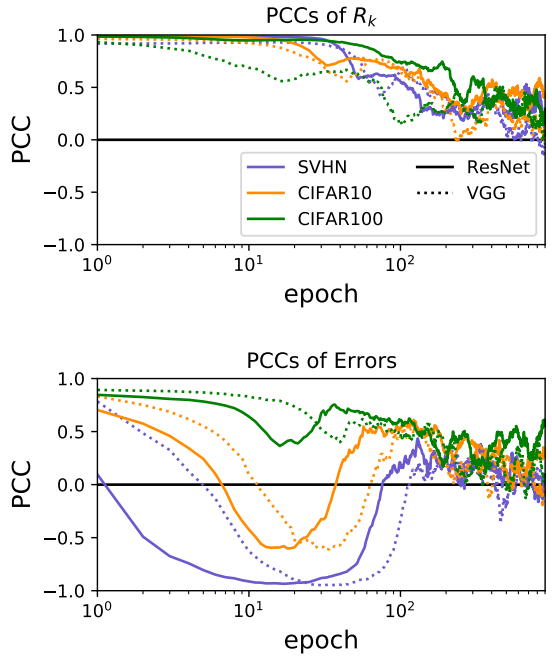


Figure 5: Short-time PCCs of R_k and errors. The colors represent different datasets, whereas the styles of lines indicate different categories of architectures, e.g., green dashed line represents VGG on CIFAR100. The short-time PCC of R_k is averaged over $k = 1, 2..., 64$. The horizontal axis is shown in logarithmic scale.

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.

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 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.

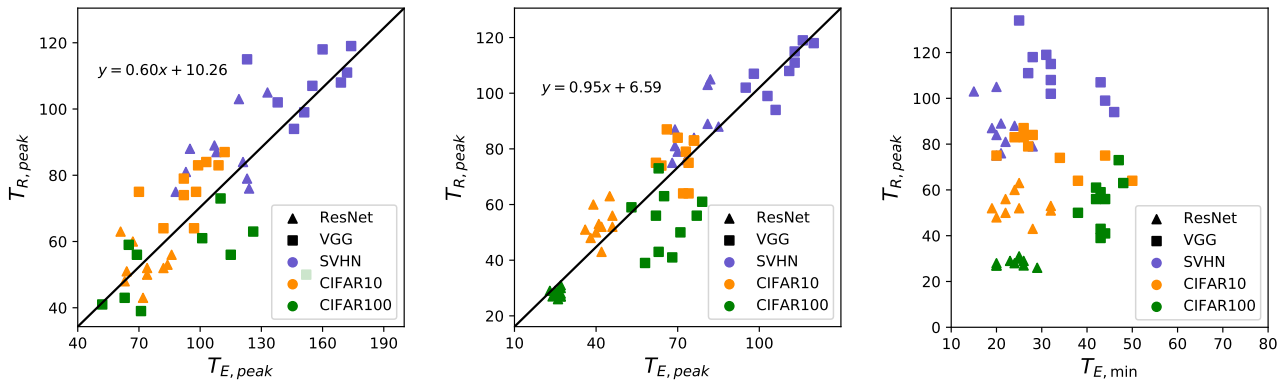
Let $R_{k,t}$ denote R_k at the t-th epoch ($R_{k,t}$ was smoothed by mean value filtering with a window length of 10 epochs), and E_t the corresponding test error. Similar to early stopping, we searched the peak of E_t and $R_t = \sum_k \alpha_k R_{k,t}$ with the patience of 30 epochs, where α_k is 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 spectrums 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.

$T_{R,\text{peak}}$ 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.

Discussions

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 point" of the training epoch does not change when the training and validation sets are combined; and, 2) it is time-consuming to train the mod-



(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. $T_{E,\text{min}}$. 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.

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

Related Work

There are a number of 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, Montufar, and Bengio 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. 2019; Xu, Zhang, and Xiao 2019). For spectral bias, some studies investigated the convergence rate of different frequencies of learned function (Ronen et al. 2019; Cao et al. 2019), but they were very specific, e.g., using DNNs with infinity width (Jacot, Gabriel, and Hongler 2018) or synthetic datasets. Moreover, all of them suggested that the process of learning bias is monotonic, which is different from our findings.

Double Descent There are 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 is based on the epoch-wise double descent and provides a novel perspective to analyzing it.

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 dose not always hold, and consequently, the low-frequency components may be sufficient to memorize the noise in the training set. We also illustrated that unlike errors, the spectrum shows remarkable consistency on the training sets, perturbed 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:

- Analyze the spectrum of DNNs in other learning tasks, e.g., natural language processing, speech recognition and so on.
- Find a new metric which can be easily calculated on the training set, but effective enough to indicate the start of overfitting.

- Explore the role that SGD and skip connection play in the spectrums of DNNs.

Acknowledgements

This work was supported by 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, R.; Basu, A.; Mianjy, P.; and Mukherjee, A. 2018. Understanding deep neural networks with rectified linear units. In *Proc. Int'l Conf. on Learning Representations*. Vancouver, Canada.

Arpit, D.; Jastrzebski, S.; Ballas, N.; Krueger, D.; Bengio, E.; Kanwal, M. S.; Maharaj, T.; Fischer, A.; Courville, A.; Bengio, Y.; and Lacoste-Julien, S. 2017. A closer look at memorization in deep networks. In *Proc. 34th Int'l Conf. on Machine Learning*, volume 70, 233–242. Sydney, Australia.

Belkin, M.; Hsu, D.; Ma, S.; and Mandal, S. 2018. Reconciling modern machine learning and the bias-variance trade-off. *CoRR* abs/1812.11118.

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

Geiger, M.; Jacot, A.; Spigler, S.; Gabriel, F.; Sagun, L.; dAscoli, S.; Biroli, G.; Hongler, C.; and Wyart, M. 2020. Scaling description of generalization with number of parameters in deep learning. *Journal of Statistical Mechanics: Theory and Experiment* 2020(2): 023401.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 770–778. Las Vegas, NV.

Jacot, A.; Gabriel, F.; and Hongler, C. 2018. Neural tangent kernel: Convergence and generalization in neural networks. In *Proc. Advances in Neural Information Processing Systems*, 8571–8580. Montreal, Canada.

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 *Proc. Advances in Neural Information Processing Systems*, 3491–3501. Vancouver, Canada.

Kingma, D. P.; and Ba, J. 2014. Adam: A Method for Stochastic Optimization. In *Proc. Int'l Conf. on Learning Representations*. Banff, Canada.

Krizhevsky, A. 2009. Learning multiple layers of features from tiny images URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Proc. Advances in Neural Information Processing Systems*, 1097–1105. Lake Tahoe, NE.

Li, H.; Xu, Z.; Taylor, G.; Studer, C.; and Goldstein, T. 2018. Visualizing the loss landscape of neural nets. In *Proc. Advances in Neural Information Processing Systems*, 6389–6399. Montreal, Canada.

Montufar, G. F.; Pascanu, R.; Cho, K.; and Bengio, Y. 2014. On the number of linear regions of deep neural networks. In *Proc. Advances in Neural Information Processing Systems*, 2924–2932. Montreal, Canada.

Nakkiran, P.; Kaplun, G.; Bansal, Y.; Yang, T.; Barak, B.; and Sutskever, I. 2020. Deep double descent: Where bigger models and more data hurt. In *Proc. Int'l Conf. on Learning Representations*. Addis Ababa, Ethiopia.

Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; and Ng, A. Y. 2011. Reading Digits in Natural Images with Unsupervised Feature Learning. In *Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*. Granada, Spain.

Northcutt, C. G.; Jiang, L.; and Chuang, I. L. 2019. Confident Learning: Estimating Uncertainty in Dataset Labels. *CoRR* abs/1911.00068.

Pascanu, R.; Montufar, G.; and Bengio, Y. 2014. On the number of response regions of deep feed forward networks with piece-wise linear activations. *CoRR* abs/1312.6098.

Poole, B.; Lahiri, S.; Raghu, M.; Sohl-Dickstein, J.; and Ganguli, S. 2016. Exponential expressivity in deep neural networks through transient chaos. In *Proc. Advances in Neural Information Processing Systems*, 3360–3368. Barcelona, Spain.

Rahaman, N.; Baratin, A.; Arpit, D.; Draxler, F.; Lin, M.; Hamprecht, F.; Bengio, Y.; and Courville, A. 2019. On the Spectral Bias of Neural Networks. In *Proc. 36th Int'l Conf. on Machine Learning*, 5301–5310. Long Beach, CA.

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

Simonyan, K.; and Zisserman, A. 2015. Very deep convolutional networks for large-scale image recognition. In *Proc. Int'l Conf. on Learning Representations*. San Diego, CA.

Vapnik, V. N. 1999. An overview of statistical learning theory. *IEEE Trans. on Neural Networks* 10(5): 988–999.

Xu, Z.-Q. J.; Zhang, Y.; Luo, T.; Xiao, Y.; and Ma, Z. 2019. Frequency principle: Fourier analysis sheds light on deep neural networks. *CoRR* abs/1901.06523.

Xu, Z.-Q. J.; Zhang, Y.; and Xiao, Y. 2019. Training behavior of deep neural network in frequency domain. In *Proc. Int'l Conf. on Neural Information Processing*, 264–274. Sydney, Australia.

Yang, Z.; Yu, Y.; You, C.; Steinhardt, J.; and Ma, Y. 2020. Rethinking Bias-Variance Trade-off for Generalization of Neural Networks. *CoRR* abs/2002.11328.

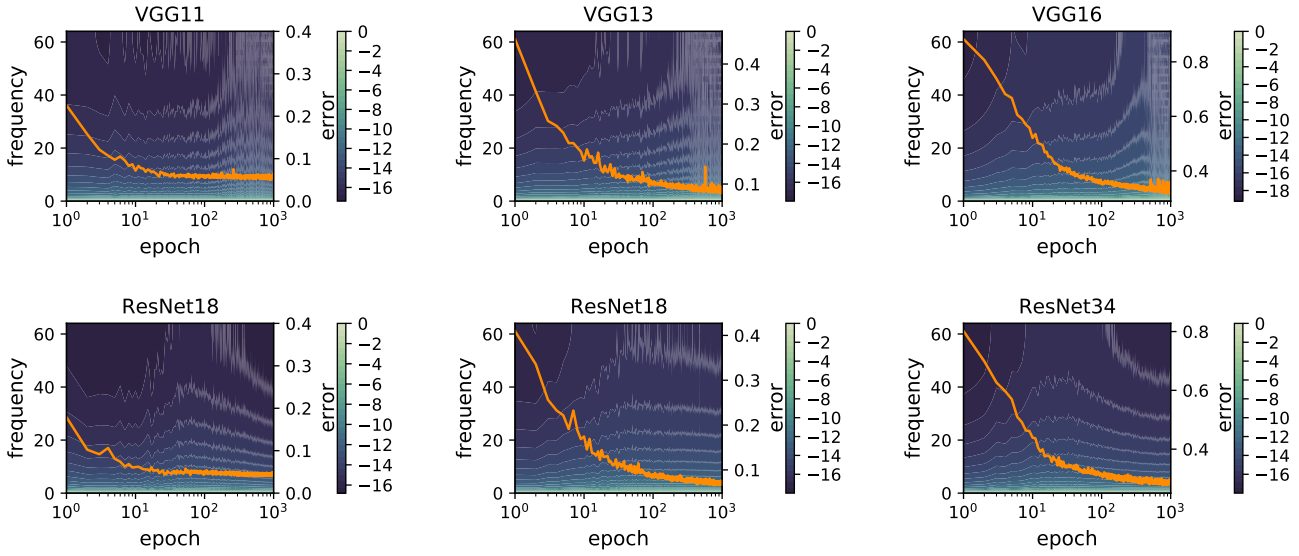
Zagoruyko, S.; and Komodakis, N. 2016. Wide residual networks. *CoRR* abs/1605.07146.

Zhang, C.; Bengio, S.; Hardt, M.; Mozer, M. C.; and Singer, Y. 2020. Identity crisis: Memorization and generalization under extreme overparameterization. In *Proc. Int'l Conf. on Learning Representations*. Addis Ababa, Ethiopia.

Zhang, C.; Bengio, S.; Hardt, M.; Recht, B.; and Vinyals, O. 2017. Understanding deep learning requires rethinking generalization. In *Proc. Int'l Conf. on Learning Representations*. Toulon, France.

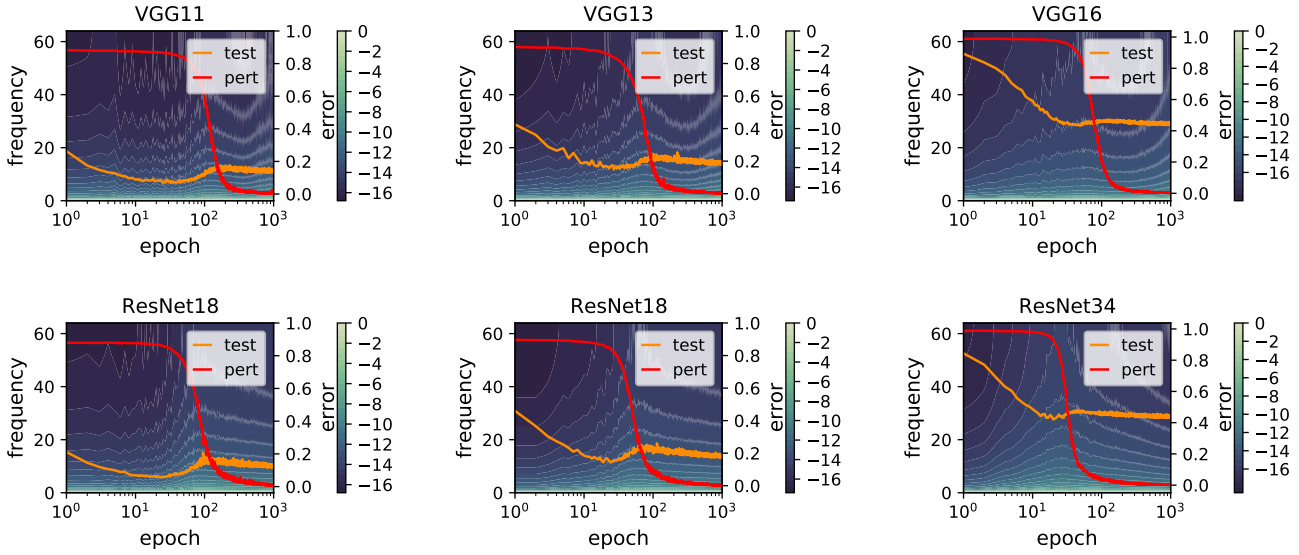
Zhang, X.; and Wu, D. 2020. Empirical Studies on the Properties of Linear Regions in Deep Neural Networks. In *Proc. Int'l Conf. on Learning Representations*. Addis Ababa, Ethiopia.

A R_k with Different Levels of Label Noise



(a) SVHN (b) CIFAR10 (c) CIFAR100

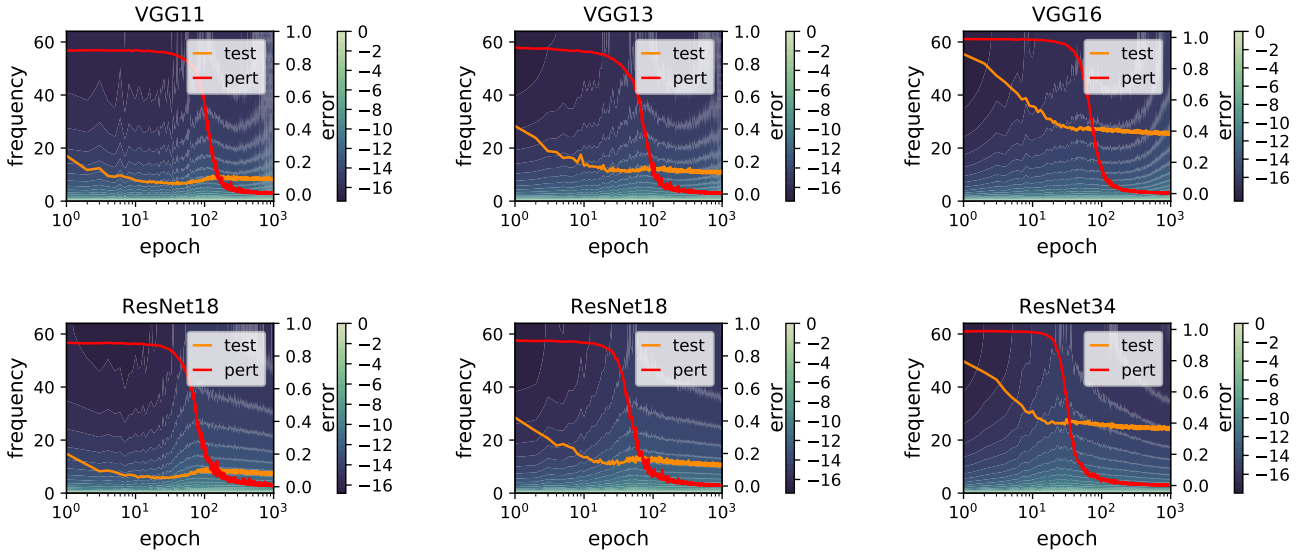
Figure 1: Models trained on SVHN, CIFAR10 and CIFAR100 with **0%** label noise. The heat map with contour lines depicts R_k which is calculated on the training set, where k denotes the "frequency". The x-axis is shown in logarithmic scale.



(a) SVHN (b) CIFAR10 (c) CIFAR100

Figure 2: Models trained on SVHN, CIFAR10 and CIFAR100 with **20%** label noise. The heat map with contour lines depicts R_k which is calculated on the training set, where k denotes the "frequency". The x-axis is shown in logarithmic scale.

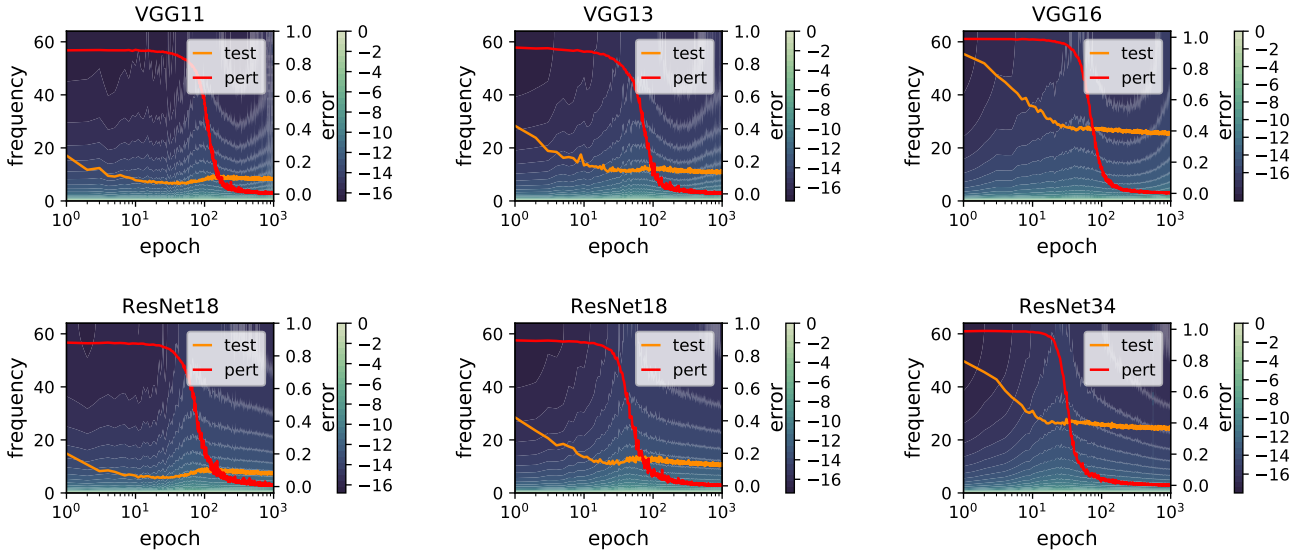
B R_k on Test Sets



(a) SVHN (b) CIFAR10 (c) CIFAR100

Figure 3: Models trained on SVHN, CIFAR10 and CIFAR100 with 10% label noise. The heat map with contour lines depicts R_k, which is calculated on the **test set**, where k denotes the "frequency". The x-axis is shown in logarithmic scale.

C R_k on Perturbed Sets



(a) SVHN (b) CIFAR10 (c) CIFAR100

Figure 4: Models trained on SVHN, CIFAR10 and CIFAR100 with 10% label noise. The heat map with contour lines depicts R_k which is calculated on the **perturbed set**, where k denotes the "frequency". The x-axis is shown in logarithmic scale.