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

Xiao Zhang & Dongrui Wu
Ministry of Education Key Laboratory of Image Processing and Intelligent Control,
School of Artificial Intelligence and Automation,
Huazhong University of Science and Technology, Wuhan, China
{xiao_zhang,drwu}@hust.edu.cn

Abstract

Over-parameterized deep neural networks (DNNs) with sufficient capacity to memorize random noise can achieve excellent generalization performance on normal datasets, 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 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 it is not true: 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.

1 Introduction

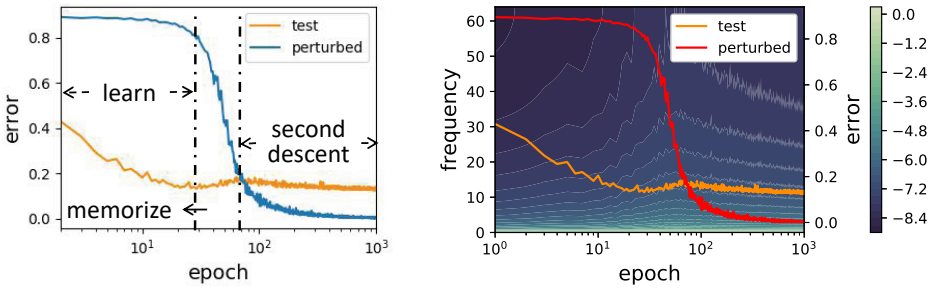
The bias-variance trade-off in classical learning theory suggests that models with large capacity to minimize the empirical risk to almost zero usually yield poor generalization performance [1]. However, this is not the case of modern deep neural networks (DNNs): Zhang *et al.* (2017) [2] showed that 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 [3–7].

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) [8] claimed that DNNs learn patterns at 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) [9] 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 [10–12]. 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.

Preprint. Under review.

All above the findings 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 [13]. 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 (see the *second descent* phase in Figure 1(a)), the high-frequency components somehow begin to diminish (see Figure 1(b)). In other words, low-frequency components may be sufficient to memorize the noise.



(a) (b)

Figure 1: ResNet18 trained on CIFAR10 with 10% label noise (Adam [14] with learning-rate 1e-4 for 1,000 epochs). The x-axis is shown in logarithmic scale. (a) Training error on the perturbed part of the training set and the corresponding test error. The vertical dash-dotted lines indicate early stopping and the start of the second descent. (b) The heat map with contour lines showing the ratio (in logarithmic scale) of the frequency energy to the basic frequency energy at every epoch (Section 2.2). The heat map is synchronized with the test error and the perturbed error.

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 on 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: Section 2 presents our analysis of the spectrums of DNNs. Section 3 shows that we can monitor the test behaviors from the training process according to the spectrum. Section 4 summarizes some related works. Section 5 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.

2 Frequency Components across the Training Procedure

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.

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) [10] 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) [11] 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 $\mathbf{x} \sim \mathcal{D}$, a normalized random direction by $\mathbf{v}_x$, the c-th logit output of a DNN by $f_c(\mathbf{x})$, where $c \in \{1, 2, ..., C\}$ and C is the number of classes. We evenly sample N points from $[\mathbf{x} - h\mathbf{v}_x, \mathbf{x} + h\mathbf{v}_x]$ to perform Fourier transform, where h bounds the area. The Fourier transform of $f_c(\mathbf{x})$ is then:

$$\tilde{f}_{c,\mathbf{x}}(k) = \sum_{n=0}^{N} f_c(\mathbf{x} + \frac{2n - N}{N} h\mathbf{v}_x) e^{-i2\pi \frac{n}{N} k}. \tag{1}$$

We use the logit outputs instead of the probabilities so that the spectrum is irrelevant to the rescaling of the DNN parameters. That is, when the weights of the last layer are multiplied by α (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:

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

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

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 [13]. We here consider two architectures[1] (VGG [4] and ResNet [5]) on three image datasets (SVHN [15], CIFAR10 and CIFAR100 [16]). We randomly shuffled 10% labels of the training set to strengthen double descent and denoted the perturbed part by *perturbed set*. The batch-size was set to 128, and we utilized the Adam optimizer [14] with the learning rate 0.0001 to train the models for $1,000$ epochs with data augmentation.

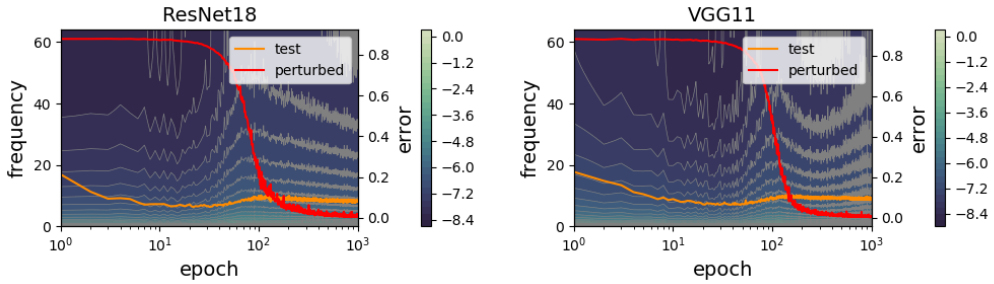
In order to better compare the spectrum along the training process, we normalized A_k according to the basic frequency energy A_0:

$$R_k = \log\left(A_k / A_0\right). \tag{3}$$

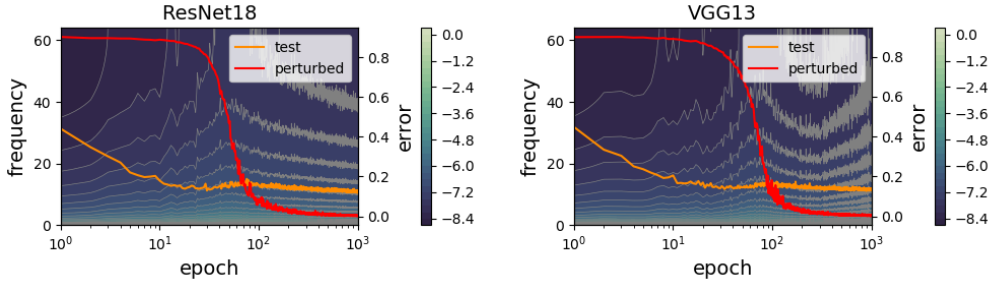
[1] Adapted from `https://github.com/kuangliu/pytorch-cifar`

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

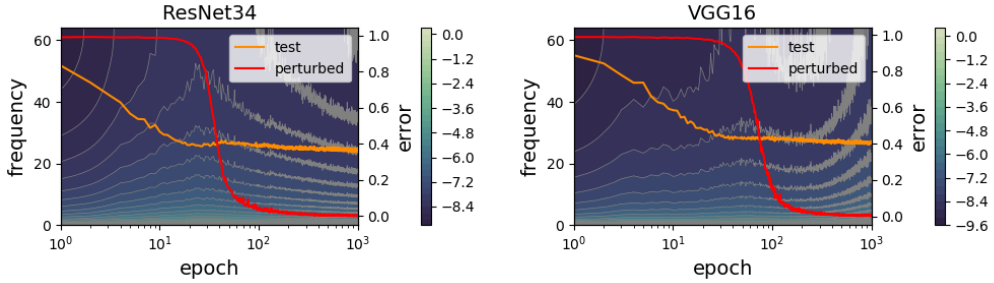
The spectrums and learning curves are shown in Figure 2. From the error curves, we can observe that the test error decreases very quickly at the beginning of the training, whereas the error on the perturbed set remains high, suggesting that the models effectively learn the patterns of the data in this period. However, as the training goes on, the error on the perturbed set deceases 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 [13].



(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 x-axis is shown in logarithmic scale.

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-

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

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 deceases, 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.

2.3 Discussion

In this subsection, we discuss the role the label noise plays in the training process, and show that it is one of the causes of the high-frequency components.

To confirm that, we trained models on clean data under the same training setup, and compared their spectrums with models trained with 10% label noise. Figure 3 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 C presents other spectrums of models trained without the label noise (one may also observe a peak of R_k for ResNet18 on SVHN and ResNet34 on CIFAR100, but the peaks become sharper when the label noise is applied.).

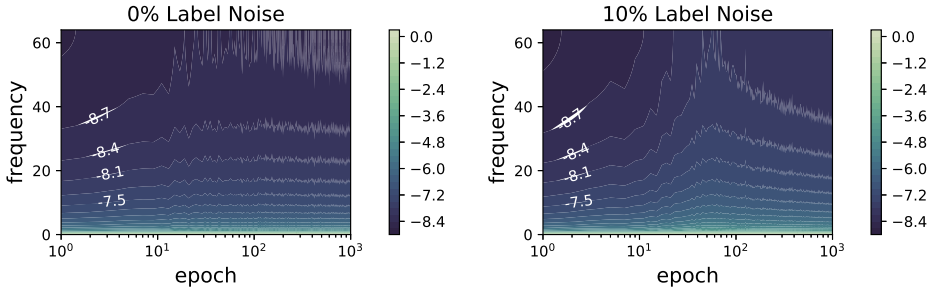


Figure 3: ResNet18 trained on CIFAR10 with 0% and 10% label noise. The white numbers indiciate the values of contour lines in the heat map R_k. The x-axis is shown in logarithmic scale.

It should be noted that the label noise is a fundamental phenomenon naturally existing in a wide range of different forms [17], 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. However, this explanation may lead to a new question:

Why do 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 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. Figure 4 shows the spectrum calculated only on the perturbed set. We can still observe the diminishment of the high-frequency components (Appendix D gives more examples). 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. But this still cannot explain why the high-frequency components are introduced at first. We will systematically study the phenomenon in our future research.

Another interesting finding is that the architectures of DNNs also 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. It is well-known that the architecture of DNNs introduces strong priors to training and results in some inductive biases, but how

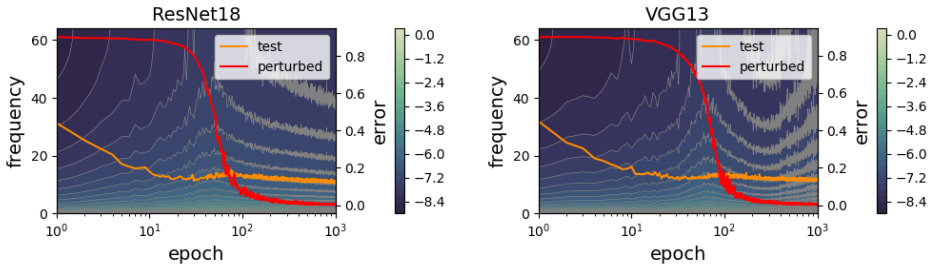


Figure 4: R_k on the perturbed set, synchronized with the test and perturbed errors. ResNet18 and VGG13 are trained on CIFAR10 with 10% label noise. The x-axis is shown in logarithmic scale.

it influences the frequency components during training is still an open problem, which may inspire the design of new architectures.

3 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.

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 E and shows no significant difference with R_k calculated on the training set (See Figure 2). We calculated Pearson correlation coefficients (PCCs) of training and testing errors, and R_k for $k = 1, 2, ..., 64$ as well. As shown in Figure 5, R_k shows better relevance on the training and test sets than errors. Appendix A also gives the short-time PCCs, which depicts how the PCCs vary w.r.t. the training epochs. All these results suggest that R_k is a metric changing consistently on the training and test sets.

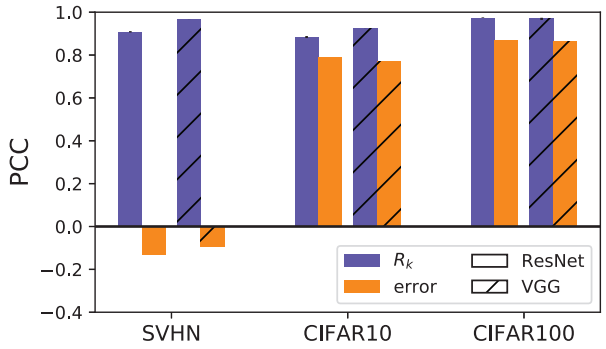


Figure 5: PCCs of errors (in orange) and R_k (in purple). The stripes represent different types of DNNs. The average PCCs of $R_k(k = 1, 2, ..., 64)$ is presented with the error bar which indicates the standard deviations (very small and hard to be seen).

From the above observation and the discussion in Section 2, 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.

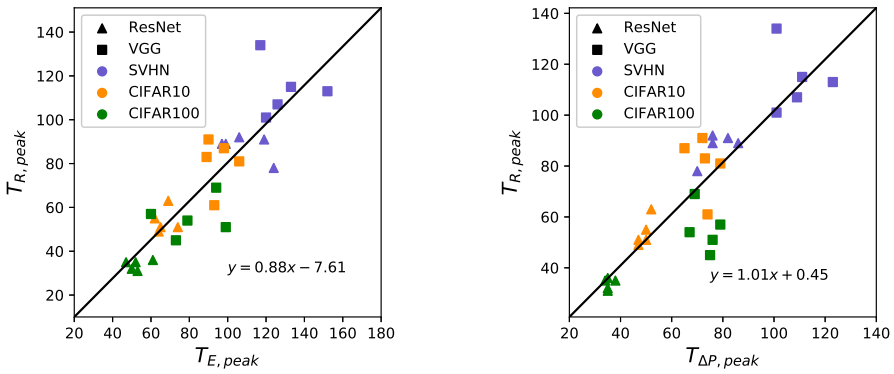
3.2 Use R_k to Monitor the Second Descent of the Test Error

We have verified the consistency of R_k on training and test sets. To monitor the test curves, the metric should also have 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, 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 equaling T ($T = 30$ in our experiments), 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 for five runs (SVHN: ResNet18 and VGG11; CIFAR10: ResNet18 and VGG13 ; CIFAR100: ResNet34 and VGG16), and explored the relation 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.



(a) (b)

Figure 6: (a) $T_{R,\text{peak}}$ w.r.t. $T_{E,\text{peak}}$; (b) $T_{R,\text{peak}}$ w.r.t. $T_{\Delta P,\text{peak}}$. 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.

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

4 Related Work

There are a number of works related to the themes of this article:

Generalization and Memorization Over-parameterized DNNs are believed to have large expressivity, usually measured by the number of linear regions in the input space [18–22]. However, it cannot explain why a DNN, whose capacity is large enough to fit random noise, still has low variance on normal datasets [2]. Arpit *et al.* (2017) [8] examined the role of memorization in deep learning and showed its connections to the model capacity and generalization. Zhang *et al.* (2020) [23] 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 the solutions, e.g., from learning patterns to memorizing noise [8], from simple to complex [9], or from low frequency to high frequency [10–12]. For spectral bias, some works studied the convergence rate of different frequencies of learned function [24, 25], but they were very specific, e.g., using DNNs with infinity width [26] 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 [13]. The former is a general phenomenon observed by many studies [27–29], whereas the latter was proposed recently, inspired by a unified notion of "Effective Model Complexity" [13]. Our work is based on the epoch-wise double descent and provides a novel perspective to analyzing it.

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 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 simply 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.

Acknowledgments and Disclosure of Funding

This research was supported by the National Natural Science Foundation of China Grant 61873321 and Hubei Technology Innovation Platform Grant 2019AEA171.

References

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

[2] 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.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. Advances in Neural Information Processing Systems*, pages 1097–1105, Lake Tahoe, NE, December 2012.

[4] 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.

[5] 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.

[6] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *CoRR*, abs/1605.07146, 2016.

[7] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Proc. Advances in Neural Information Processing Systems*, pages 6389–6399, Montreal, Canada, December 2018.

[8] Devansh Arpit, Stanisław Jastrzębski, 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. Int'l Conf. on Learning Representations*, Banff, Canada, April 2014.

[15] 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.

[16] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. URL `https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf`.

[17] Curtis G Northcutt, Lu Jiang, and Isaac L Chuang. Confident learning: Estimating uncertainty in dataset labels. *CoRR*, abs/1911.00068, 2019.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] 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.

[25] 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.

[26] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Proc. Advances in Neural Information Processing Systems*, pages 8571–8580, Montreal, Canada, December 2018.

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

[28] 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.

[29] 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.

A Short-Time Pearson Correlation Coefficient

The original Pearson correlation coefficient (PCC) of two series $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ is calculated as:

$$p(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{n}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{y}_i - \bar{\mathbf{y}})}{\sqrt{\sum_{i=1}^{n}(\mathbf{x}_i - \bar{\mathbf{x}})^2 \sum_{i=1}^{n}(\mathbf{y}_i - \bar{\mathbf{y}})^2}}, \tag{4}$$

where $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$ are the means. The PCC can show the relevance of two series, but lacks of accurate local description if the series are very long. Therefore, we use short-time PCC to show the PCCs w.r.t. the training epoch t, which is the PCC of series in a sliding window of length l (set to 100 in our experiments):

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

Figure 7 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 deceases 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.

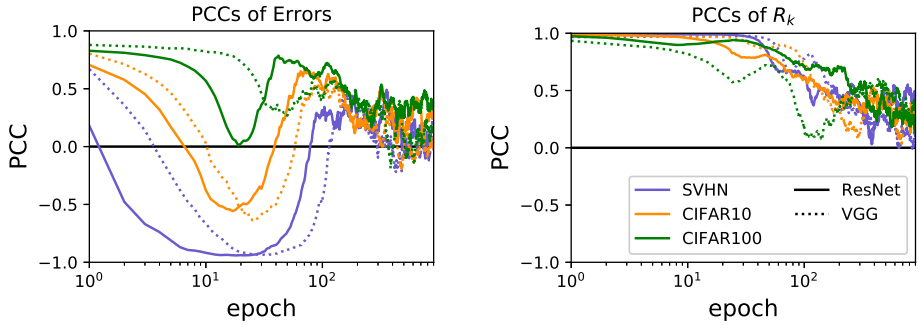


Figure 7: Short-time PCCs of errors (left) and R_k (right). The colors represent different datasets, whereas the styles of lines indicate different categories of architectures, e.g., green dashed line represents VGG on CIFAR100. The short-time PCC of R_k is averaged over $k = \{1, 2..., 64\}$. The x-axis is shown in logarithmic scale.

B $T_{R,\text{peak}}$ w.r.t. $T_{E,\min}$

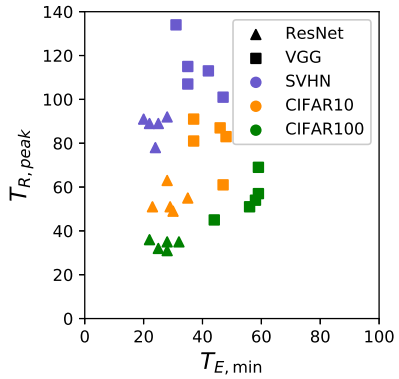
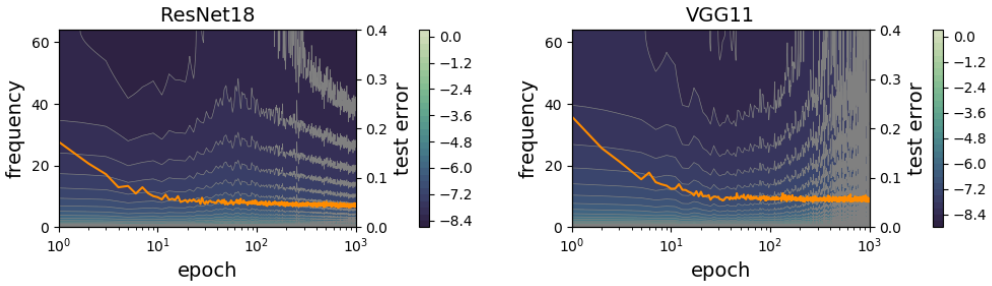
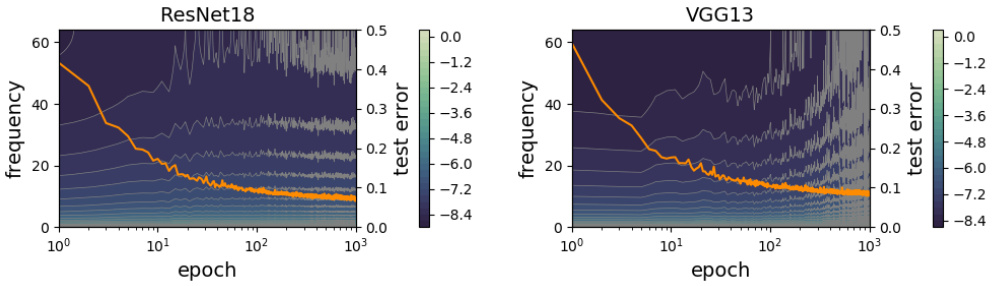


Figure 8: $T_{R,\text{peak}}$ w.r.t. $T_{E,\min}$. The colors represent different datasets, whereas the shapes indicate different categories of DNNs, e.g., green squares represent VGG on CIFAR100. Each experiment was repeated five times.

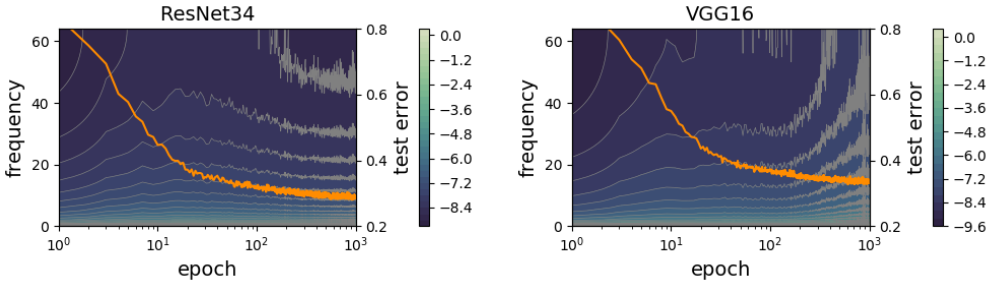
C R_k with 0% Label Noise



(a) SVHN



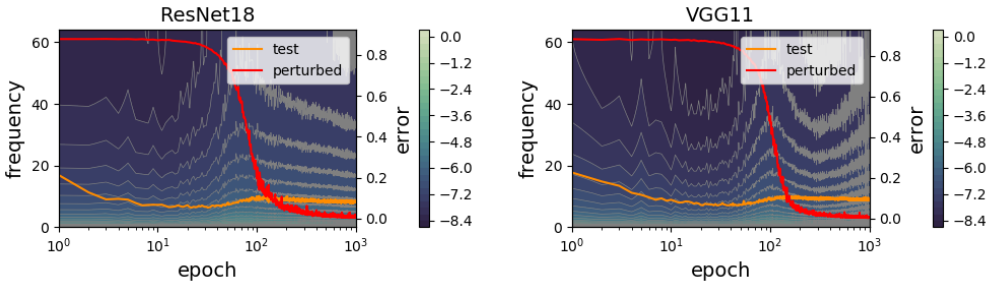
(b) CIFAR10



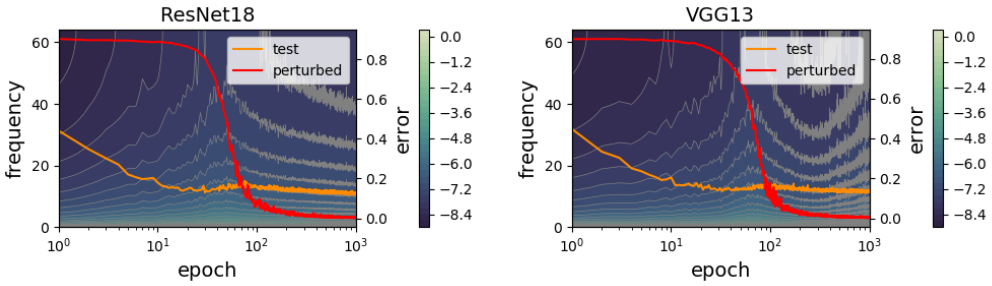
(c) CIFAR100

Figure 9: ResNet and VGG trained on SVHN, CIFAR10 and CIFAR100 with no label noise. The heat map with contour lines depicts R_k, where k denotes the "frequency". The x-axis is shown in logarithmic scale.

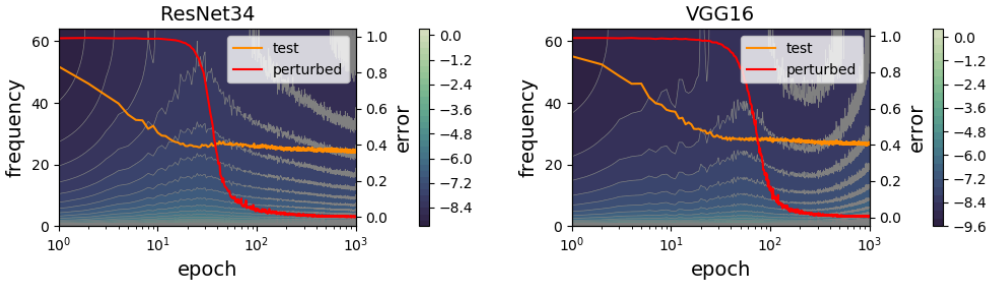
D R_k on Perturbed Sets



(a) SVHN



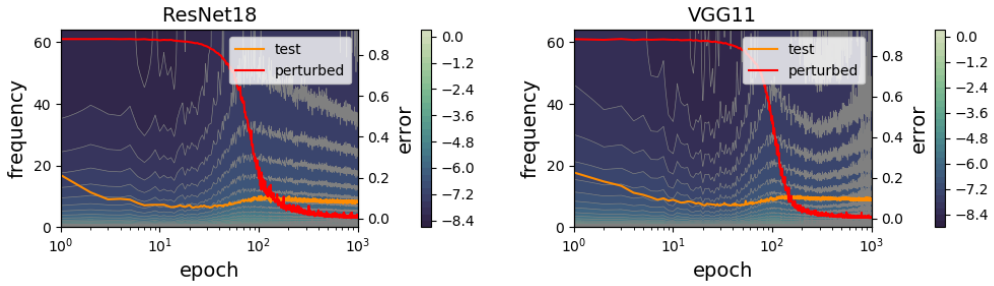
(b) CIFAR10



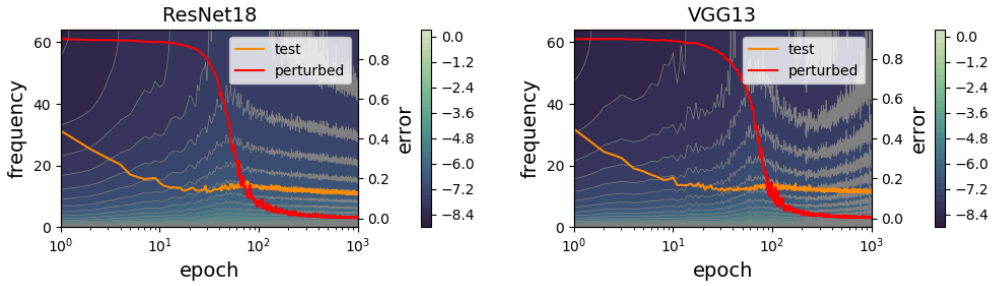
(c) CIFAR100

Figure 10: ResNet and VGG 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.

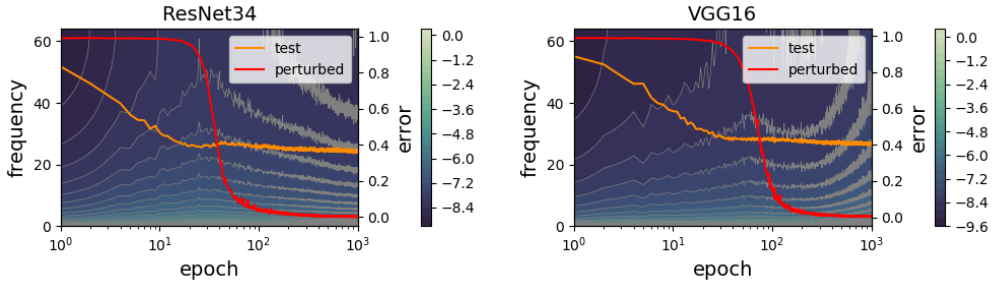
E R_k on Test Sets



(a) SVHN



(b) CIFAR10



(c) CIFAR100

Figure 11: ResNet and VGG 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.