# WIN THE LOTTERY TICKET VIA FOURIER ANALYSIS: FREQUENCIES GUIDED NETWORK PRUNING

*Yuzhang Shang*[1] *Bin Duan*[1] *Ziliang Zong*[2] *Liqiang Nie*[3] *Yan Yan*[1]

[1]Illinois Institute of Technology [2]Texas State University [3]Shandong University

## ABSTRACT

With the remarkable success of deep learning recently, efficient network compression algorithms are urgently demanded for releasing the potential computational power of edge devices, such as smartphones or tablets. However, optimal network pruning is a non-trivial task which mathematically is an NP-hard problem. Previous researchers explain training a pruned network as buying a lottery ticket. In this paper, we investigate the Magnitude-Based Pruning (MBP) scheme and analyze it from a novel perspective through Fourier analysis on the deep learning model to guide model designation. Besides explaining the generalization ability of MBP using Fourier transform, we also propose a novel two-stage pruning approach, where one stage is to obtain the topological structure of the pruned network and the other stage is to retrain the pruned network to recover the capacity using knowledge distillation from lower to higher on the frequency domain. Extensive experiments on CIFAR-10 and CIFAR-100 demonstrate the superiority of our novel Fourier analysis based MBP compared to other traditional MBP algorithms.

***Index Terms***— Network Compression, Network Pruning, Unstructured Pruning, Fourier Analysis

## 1. INTRODUCTION

Over the past decade, utilizing deep learning to tackle intricate tasks, which are barely solved using shallow machine learning techniques solely, has achieved extremely outstanding performance in various domains of computer vision [1, 2, 3], information retrieval [4, 5] and multi-modal learning [6, 7, 8]. However, with the drastic increase of neural network parameters, computational resources are often short-handed. This hinders the broader deployment of deep models especially for edge devices. To solve this shortage problem, network pruning is proposed, which builds upon the prune-retrain-prune paradigm [9] and is proved to be a practical approach to reduce the computation cost of overparameterized deep neural networks (DNNs).

Following the prune-retrain-prune paradigm, MBP is proposed in [10, 11] which show that pruning weights of small magnitudes and then retraining the model without contaminating the overall accuracy where simultaneously reach a pretty low compression rate on neural network models. Frankle et al. [12] manifests that the MBP algorithm is to seek an efficient trainable subnetwork as a replacement for the original dense neural networks, where they empirically demonstrate that there exist sparse sub-networks that can be trained from scratch and perform competitively only if the initialization weights of this subnetwork are appropriate. Moreover, MBP is studied further in [13, 14] where the authors use a straight-through an estimator in the backward pass and by which they achieve the state-of-the-art for unstructured pruning in DNNs. Yet despite all of MBP methods testify the effectiveness of MBP, no rational explanation has been endowed for why we can successfully retrain the pruned networks driven from MBP.

In this paper, we investigate the MBP scheme from a novel perspective, Fourier analysis where we regard DNN as a complex function extracting abstract representations for different datasets or tasks. Whilst pruning tries to remove less important representations, measuring the importance of those representations is non-trivial. Thanks to the power of Fourier analysis which can approximate any arbitrary functions by the sum of simpler eigenfunctions on the frequency domain, we intuitively want to measure the importance of the DNN's representations with Fourier analysis. It is proved that harmonic analysis, especially Fourier analysis, can efficiently obtain a spectral perspective of a perplexing system. However, due to the complexity of DNNs, conducting Fourier transformation on them remains a problem. Lately, [15, 16, 17] and [18] study DNNs training by Fourier analysis and reveal some profound results which are: (i) DNNs tend to fit lower-frequency components first during training by gradient descent, and (ii) lower-frequency components are more robust to random perturbations of network parameters. We draw inspirations from [15, 18] and design our network pruning scheme. Additionally, we endow an explanation using Fourier analysis for why the pruned network succeeds.

By embracing network pruning with Fourier analysis, the knowledge of data can be divided into lower frequencies and higher frequencies in the frequency domain in term of the phase of training unpruned network. Motivated by above assumptions, we propose a novel two-stage pruning method **WILTON** (**WI**n the **L**ottery **T**ickets through the F**O**urier A**N**alysis: Frequencies Guided Network Pruning, using the names "Lottery Tickets" adopted from "Lottery Ticket Hypothesis" [12]). We first acquire the topological structure and initialization weights by MBP scheme from the network representing lower-frequency, and then utilize knowledge distillation to recover the capacity where we learn the pruned network from the unpruned network which contains higher-frequency components. As shown in Fig. 1, our architecture is designed in a parallel fashion to separately learn the structure and parameters of the pruned network.

In summary, our contributions can be highlighted as follows. (i) Fourier transform is introduced to the network pruning tasks to understand the generalization ability of MBP. (ii) A novel pruning method called WILTON is proposed to train the pruned neural network gradually from lower frequency to higher frequency. (iii) Our experimental results demonstrate that the proposed pruning scheme outperforms other recent MBPs on various network architectures, including ResNet [2] and VGGNet [19].

## 2. WILTON

### 2.1. Fourier Analysis of Neural Networks

For general purpose, we only take the networks with rectified linear unit (ReLU) activations as an example in this section.

We define the ReLU network with $L$ layers of widths $d_1, \cdots d_L (d = \sum_{k=1}^{L} d_k)$ and a single output neuron a scalar function$f : \mathbb{R}^d \longmapsto \mathbb{R}$ :

$$f(\mathbf{x}) = (T^{(L)} \circ \sigma \circ T^{(L-1)} \circ \cdots \circ \sigma \circ T^{(1)})(\mathbf{x}) \quad (1)$$

where each $T^{(k)} : \mathbb{R}^{d_{k-1}} \longmapsto \mathbb{R}^{d_k}$ is an affine function and $\sigma(\mathbf{u}) = max(0, u_i)$ denotes the ReLU activation function acting elementwise on a vector $\mathbf{u} = (u_1, \cdots u_n)$. In the standard basis, $T^{(k)}(\mathbf{u}) = W^k\mathbf{u} + \mathbf{b}^k$, the $W^k$ and $\mathbf{b}^k$ stand for the weight matrix and bias vector of the network's $k$-th layer, respectively. For clarity and brevity, we discard the bias term of the network, then the network can be represented as:

$$f(W^1, \cdots, W^L; \mathbf{x}) \quad (2)$$

The Fourier transform of the ReLU network is fairly an intricate mathematical problem. In [20] and [18], the authors develop an elegant procedure for evaluating it in arbitrary dimensions via a recursive application of Stokes theorem. We study the structure of ReLU networks in terms of their Fourier representation, $f(\mathbf{x}) := (2\pi)^{d/2} \int \tilde{f}(\mathbf{k}) e^{i\mathbf{k}\cdot\mathbf{x}} \mathbf{dk}$, where $\tilde{f}(\mathbf{k}) := \int f(\mathbf{x}) e^{i\mathbf{k}\cdot\mathbf{x}} \mathbf{dx}$ is the Fourier transform. We skip the formula derivation process and only present the main corollary here.

The Fourier components of the ReLU network $\widetilde{f_\theta}$ with parameters $\theta$ is given by the function:

$$\widetilde{f_\theta}(\mathbf{k}) = \sum_{n=0}^{d} \frac{C_n(\theta, \mathbf{k}) 1_{H_n^\theta}(\mathbf{k})}{k^{(n+1)}} \quad (3)$$

where $C_n(\theta, \cdot) : \mathbb{R}^d \longmapsto \mathbb{C}$, $H_n^\theta$ is the union of n-dimensional subspaces and $1_{H_n^\theta}$ is the indicator over $H_n^\theta$. Note that Eq.3 applies to general ReLU networks with arbitrary width and depth.

The numerator in Eq.3 is bounded by $O(L_f)$, where $L_f$ is the Lipschitz constant of the network $f(\mathbf{x})$. Further the Lipschitz constant $L_f$ can be bounded by:

$$L_f \leq \prod_{k=1}^{L} \|W^k\| \leq \|\theta\|_\infty^L \sqrt{d} \prod_{k=1}^{L} d_k \quad (4)$$

where $\|\cdot\|$ is the spectral norm of matrix and $\|\cdot\|_\infty$ is the max norm of vector, $d_k$ is the number of units in $k$-th layer, $d = \sum_{k=1}^{L} d_k$ and $\theta$ is the concatenated vector of the network's whole parameters.

### 2.2. WILTON Pruning Architecture

For mathematical formulation, we use $M^k \in \{0,1\}^{d_{k-1} \times d_k}, (k = 1, \cdots L)$ to denote a mask on $W^k$ and define $W_p^k, (k = 1, \cdots L)$ as the parameters of pruned network. Note that the matrix $M^k, W^k$ and $W_p^k, (k = 1, \cdots L)$ have same shape. Then the network pruned network process can be defined by the matrix element-wise (Hadamard) product of mask $M^k$ and original weights $W^k$: $W_p^k = M^k \odot W^k, (k = 1, \cdots L)$. The pruned network can be represented as:

$$f(W_p^1, \cdots, W_p^L; \mathbf{x}) = f(M^1 \odot W^1, \cdots, M^L \odot W^L; \mathbf{x}). \quad (5)$$

And then we can define the sparsity of pruned network $s = \frac{\sum_{k=1}^{L} \|M^k\|_0}{d}$. A pruning strategy should yield a pruned network, which can maximize prediction accuracy given an overall parameter budget.

In the view of knowledge distillation, [21, 22, 23] demonstrate that the distilled knowledge can be considered as the flow between layers of network. As the neural network maps from the input space to the output space through many layers sequentially, by simplifying Eq.2, we assume the distilled knowledge to be transferred in terms of flow between layers, which is calculated by computing the Matrix-chain multiplication from successive layers' matrix:

$$\bar{\mathbf{W}} = \prod_{k=1}^{L} W^k. \quad (6)$$

Hereto, we define $\bar{\mathbf{W}}$ an important indicator to measure two networks' similarity. So, this indicator of the pruned and unpruned network should be close. In other word, after pruning the change form $\bar{\mathbf{W}}$ to $\bar{\mathbf{W}}_p = \prod_{k=1}^{L} W_p^k$ should be stable. And based on Eq.4, we have an approach to quantitatively analyze this term.

We obtain an upper bound of $\|\bar{\mathbf{W}}\|$:

$$\|\bar{\mathbf{W}}\| = \|\prod_{k=1}^{L} W^k\| \leq \prod_{k=1}^{L} \|W^k\|. \quad (7)$$

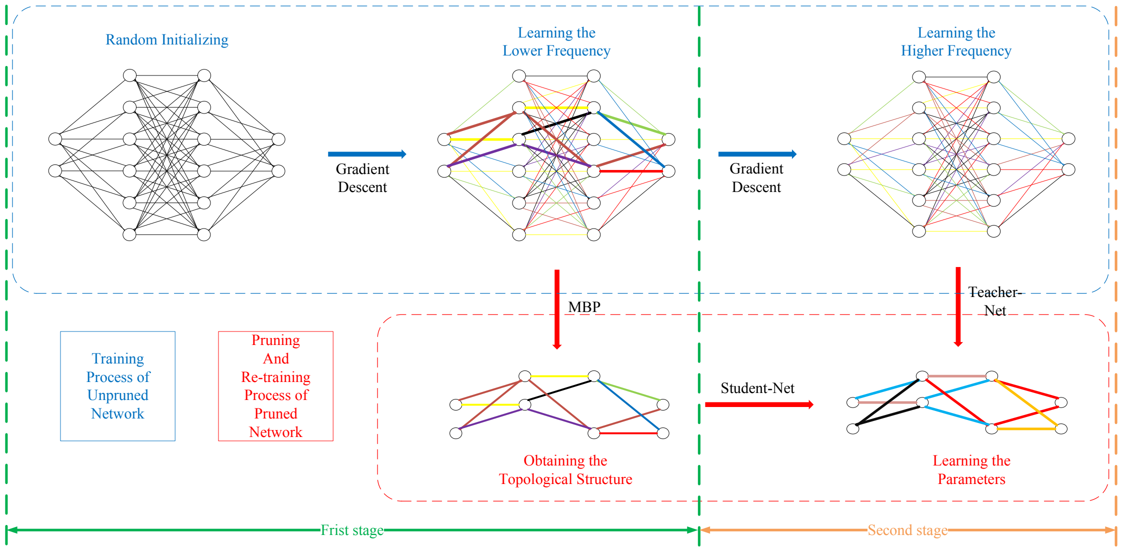
Random Initializing
Learning the Lower Frequency
Learning the Higher Frequency
Gradient Descent
Gradient Descent
MBP
Teacher-Net
Training Process of Unpruned Network
Pruning And Re-training Process of Pruned Network
Student-Net
Obtaining the Topological Structure
Learning the Parameters
Frist stage
Second stage

**Fig. 1**: The architecture of WILTON. The upper line demonstrates that the knowledge of data is divided into lower frequencies and higher frequencies in terms of the phase of training unpruned network by gradient descent. The bottom line shows that the two-stage pruning scheme firstly learns the topology structure from lower frequency and then use knowledge distillation to transfer the higher frequency to the pruned network.

By combining Eq.4 and Eq.7, we have:

$$\|\bar{\mathbf{W}}\| \leq \prod_{k=1}^{L} \|W^k\| \leq \|\theta\|_\infty^L \sqrt{d} \prod_{k=1}^{L} d_k. \tag{8}$$

Therefore we only have to focus on the last term of Eq.8. After pruning the redundant weights from unpruned network, we have $\|\bar{\mathbf{W}}_p\| = \|\prod_{k=1}^{L} W_p^k\|$ of pruned neural network. The pruning scheme should have the ability to keep this term stable. Because pruning itself reduces the number of parameters of layers, which leads to the decrease of $d_k$ and $d$. Therefore, the pruning scheme must keep $\|\theta\|_\infty^L$ as big as possible. When pruning, we should prune those weights whose magnitudes are small to keep Eq.8 stable, which is also an explanation of why magnitude-based pruning methods make sense from the perspective of Fourier analysis.

Besides the explanation, we also find more intriguing properties of DNN training through Fourier analysis. In [15, 18], the authors both demonstrate that during the DNN training process, the lower frequency of data is learned first and more robust to parameters perturbation. Based on those properties, we manually unravel the lower and higher frequency knowledge of data on the frequency domain and bridge the ideology of knowledge distillation [24, 25]. We propose our two-stage pruning scheme, WILTON, where we first obtain a topological structure and initialization weights by MBP from a low-accuracy network and then distillate knowledge from the high-accuracy unpruned network into our pruned network. The architecture of proposed WILTON is shown in Fig. 1.

## 3. EXPERIMENT

**Datasets and Metrics.** CIFAR-10 and CIFAR-100 [26] are widely used image classification datasets, which consists of 60k $32 \times 32$ color images divided into 10 and 100 classes, respectively. We compare the top-1 accuracy on CIFAR datasets for our approach (WILTON), with vanilla MBP [10], LT [12] and STR [13].

**Implementation details.** We define the pruning ratio to be $r_p = s \cdot 100(\%)$, where $s$ is the sparsity of pruned network. For a given pruning ratio $r_p$, we train the unpruned network for 200 epochs and prune the unpruned network at epoch #100. We discuss this hyperparameter in Section. 3.3. After pruning, we use the knowledge distillation to train the pruned network by the standard way. Specifically, we train the models using SGD with momentum of 0.9, batch size of 512, and the weight decay rate of 0.0005, unless state otherwise. The initial learning rate is set to 0.2 and decayed by 0.1 at every 20 or 40 epochs for CIFAR-10 and CIFAR-100, respectively. Additionally, we use the standard data augmentation (i.e., random horizontal flip and translation up to 4 pixels) for both the unpruned and pruned models. We keep 10% of the training data as a validation set and use only 90% for training.

### 3.1. Results on CIFAR-10

On CIFAR-10 [26], we perform experiments on two standard network architectures – ResNet32 and VGG16 where ResNet32 contains 0.47M parameters and VGG16 consists of 15.3M parameters. We compared our method with vanilla MBP [10, 11], LT [12] and STR [13]. We also report the performance of a full network and a random pruning baseline. All the methods are compared extensively in the pruning ratio of 90%, 95%, 98%.

Overall, Table.1 shows that our method WILTON constantly outperform or evenly perform other pruning state-of-the-art methods in all pruning ratios with different network architectures. For further comparisons, we discuss the experimental results for ResNet32 and VGG16, respectively. For

**Table 1**: Test accuracy of pruned ResNet32 and VGG16 on CIFAR-10 dataset. The higher the better. 10.00±0.00 means the network cannot be trained and thus infer as random guessing.

| Pruning ratio | 90% | 95% | 98% |
|---|---|---|---|
| ResNet32(Full Network) | 92.30 | - | - |
| random | 65.12±3.89 | 18.87±2.24 | 10.00±0.00 |
| vanilla-MBP[10] | 91.17±0.06 | 87.35±0.20 | 87.64±0.28 |
| LT[12] | 92.08±0.18 | 92.78±0.04 | 91.24±0.21 |
| STR[13] | 93.22±0.11 | **92.76±0.16** | 91.48±0.10 |
| WILTON(Ours) | **93.24±0.13** | **92.79±0.11** | **91.64±0.09** |
| VGG16(Full Network) | 91.11 | - | - |
| random | 43.26±0.34 | 10.0±0.00 | 10.00±0.00 |
| vanilla-MBP[10] | 87.26±0.15 | 41.17±0.33 | 10.00±0.00 |
| LT[12] | 92.50±0.31 | **91.17±0.09** | 10.00±0.00 |
| WILTON(Ours) | **92.88±0.03** | **91.26±0.38** | 10.00±0.00 |

**Table 2**: Test accuracy of pruned network from various training period of unpruned ResNet32 with different pruning ratios.

| Epoch # / Ratio | 0 | 50 | 100 | 150 | 200 |
|---|---|---|---|---|---|
| 90% | 93.25 | 93.27 | **93.37** | 92.28 | 91.30 |
| 95% | 92.60 | **92.90** | 92.71 | 92.66 | 92.09 |
| 98% | 91.40 | **91.73** | 91.71 | 91.37 | 91.23 |

**Table 3**: Test accuracy of pruned ResNet32, ResNet56 and VGG16 on CIFAR-100 dataset. The higher the better. 1.00±0.00 means the network cannot be trained and thus infer like random guessing.

| Pruning ratio | 90% | 95% | 98% |
|---|---|---|---|
| ResNet32(Full Network) | 72.30 | - | - |
| random | 6.30±0.21 | 1.00±0.00 | 1.00±0.00 |
| vanilla-MBP[10] | 34.96±0.90 | 27.49±0.21 | 1.00±0.00 |
| LT[12] | 70.75±0.18 | 70.26±0.27 | 65.52±0.15 |
| STR[13] | **72.23±0.06** | **71.40±0.17** | 68.05±0.13 |
| WILTON(Ours) | **72.31±0.26** | 70.39±0.02 | **68.77±0.19** |
| ResNet56(Full Network) | 72.32 | - | - |
| random | 11.02±0.85 | 4.59±0.23 | 1.00±0.00 |
| vanilla-MBP[10] | 49.71±0.28 | 27.86±0.57 | 1.00±0.00 |
| LT[12] | 70.88±0.25 | 70.17±0.48 | 66.94±0.95 |
| STR[13] | **72.40±0.74** | **71.83±0.55** | 68.58±0.22 |
| WILTON(Ours) | **72.31±0.29** | **71.39±0.32** | **69.00±0.19** |
| VGG16(Full Network) | 71.09 | - | - |
| random | 29.70±0.77 | 1.00±0.00 | 1.00±0.00 |
| vanilla-MBP[10] | 65.38±0.36 | 8.27±0.54 | 1.00±0.00 |
| LT[12] | **70.49±0.63** | 39.50±0.48 | 1.00±0.00 |
| WILTON(Ours) | **70.44±0.03** | **45.29±0.60** | 1.00±0.00 |

ResNet32, as pruning ratio increases, the performances of all methods drop in different levels. However, among all methods, WILTON tends to decay slowly which shows the robustness of our method. For VGG16, the result shows the same trend as of ResNet32 – accuracy with WILTON deteriorates more slowly than other methods.

## 3.2. Results on CIFAR-100

On CIFAR-100 [26], we conduct experiment with three network architectures – ResNet32, ResNet56 and VGG16. We compare our method with vanilla MBP [10, 11], LT [12] and STR [13]. We also report the performance of a full network and a random pruning baseline. All the methods are compared in the pruning ratio of 90%, 95%, 98%.

Our method achieves competitive and even better performance for all pruning ratios on different network architectures in Table.3. For ResNet32, as pruning ratio increases, the performances of all methods drop in different levels. However, among all methods, WILTON tends to decay slowly which shows the robustness of our method. For instance, the experiment about pruning ResNet32 shows that the model pruned by WILTON retain 69% accuracy even with only 2% remaining weights. In contrast, the model pruned with vanilla-MBP lose the ability of inference completely. It proves the observation that lower-frequency components are more robust to recover the capability of the network. For ResNet56, the pruned model with only 10% parameters remaining can outperform its unpruned counterpart over 0.3%, which indicates the overparameterization of ResNet. For VGG16, the result shows the same trend as of ResNet32 – accuracy with WILTON deteriorates more slowly than other methods. While high-pruning-ratio pruning networks without shortcut structure tends to block the pathway for forward and backward propagation, the pruning of VGG16 (without shortcut) shows that the structure of VGG16 without 98% parameters is destroyed so that all methods perform randomly guessing. An intuitive explanation is that the network approximating the lower-frequency components can be represented by a sparse network, so the pruned network driven from WILTON can keep effective to a large extent.

## 3.3. Pruning Phase Evaluation and Ablation Study

WILTON is a two-stage pruning scheme, first obtaining a topological structure and initialization weights of the pruned network by MBP and then distilling knowledge from the high-accuracy unpruned network into the pruned network. On the unpruned ResNet32 training process on CIFAR-10, we prune 90% weights of the unpruned network at different epoch (#0, #50, #100, #150 and #200) and compare different pruned networks. The results are shown in Table. 2. The performance for early pruning tends to be better than late pruning where certifies the robustness of lower-frequency components. However, extreme case like pruning in epoch #0 degrades the performance since the network has not learn enough representations. That also gives us insights on how to choose the time to pruning our model such that we can recover the capacity of the unpruned model.

## 4. CONCULSION

In this paper, we propose a novel two-stage pruning method named WILTON which is an interpretable and versatile pruning approach based on Frequency domain decomposition. By separating knowledge from lower frequency to higher frequency on the frequency domain, WILTON provides a novel pathway for exploring network pruning. Finally, our proposed WILTON serves as the stepstone to understand the optimization and initialization of deep neural networks.

**Acknowledgements.** This research was partially supported by NSF CNS-1908658, NeTS-2109982 and the gift donation from Cisco. This article solely reflects the opinions and conclusions of its authors and not the funding agents.

## 5. REFERENCES

[1] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, "Deep learning," *Nature*, 2015.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[3] Yupeng Hu, Meng Liu, Xiaobin Su, Zan Gao, and Liqiang Nie, "Video moment localization via deep cross-modal hashing," *TIP*, 2021.

[4] Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, Richang Hong, and Tat-Seng Chua, "Mmgcn: Multi-modal graph convolution network for personalized recommendation of micro-video," in *ACMMM*, 2019.

[5] Yinwei Wei, Xiang Wang, Qi Li, Liqiang Nie, Yan Li, Xuanping Li, and Tat-Seng Chua, "Contrastive learning for cold-start recommendation," in *ACMMM*, 2021.

[6] Ye Zhu, Yu Wu, Yi Yang, and Yan Yan, "Saying the unseen: Video descriptions via dialog agents," *TPAMI*, 2021.

[7] Ye Zhu, Yu Wu, Yi Yang, and Yan Yan, "Describing unseen videos via multi-modal cooperative dialog agents," in *ECCV*, 2020.

[8] Yupeng Hu, Liqiang Nie, Meng Liu, Kun Wang, Yinglong Wanga, and Xiansheng Hua, "Coarse-to-fine semantic alignment for cross-modal moment localization," *TIP*, 2021.

[9] Yann LeCun, John S Denker, and Sara A Solla, "Optimal brain damage," in *NeurIPS*, 1990.

[10] Song Han, Huizi Mao, and William J Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," in *ICLR*, 2015.

[11] Michael Zhu and Suyog Gupta, "To prune, or not to prune: exploring the efficacy of pruning for model compression," *arXiv:1710.01878*, 2017.

[12] Jonathan Frankle and Michael Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," in *ICLR*, 2018.

[13] Aditya Kusupati, Vivek Ramanujan, Raghav Somani, Mitchell Wortsman, Prateek Jain, Sham Kakade, and Ali Farhadi, "Soft threshold weight reparameterization for learnable sparsity," in *ICML*, 2020.

[14] Trevor Gale, Erich Elsen, and Sara Hooker, "The state of sparsity in deep neural networks," *arXiv:1902.09574*, 2019.

[15] Zhiqin John Xu, "Understanding training and generalization in deep learning by fourier analysis," *arXiv:1808.04295*, 2018.

[16] Dong Yin, Raphael Gontijo Lopes, Jon Shlens, Ekin Dogus Cubuk, and Justin Gilmer, "A fourier perspective on model robustness in computer vision," in *NeurIPS*, 2019.

[17] Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein, "On the expressive power of deep neural networks," in *ICML*, 2017.

[18] 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 *ICML*, 2019.

[19] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv:1409.1556*, 2014.

[20] Ricardo Diaz, Quang-Nhat Le, and Sinai Robins, "Fourier transforms of polytopes, solid angle sums, and discrete volume," *arXiv:1602.08593*, 2016.

[21] Seung Hyun Lee, Dae Ha Kim, and Byung Cheol Song, "Self-supervised knowledge distillation using singular value decomposition," in *ECCV*, 2018.

[22] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim, "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning," in *CVPR*, 2017.

[23] Yuzhang Shang, Bin Duan, Ziliang Zong, Liqiang Nie, and Yan Yan, "Lipschitz continuity guided knowledge distillation," in *ICCV*, 2021.

[24] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, "Distilling the knowledge in a neural network," *arXiv:1503.02531*, 2015.

[25] Jimmy Ba and Rich Caruana, "Do deep nets really need to be deep?," in *NeurIPS*, 2014.

[26] Alex Krizhevsky, Geoffrey Hinton, et al., "Learning multiple layers of features from tiny images," *Tech Report*, 2009.