EZCrop: Energy-Zoned Channels for Robust Output Pruning

Rui Lin[1,*] Jie Ran[1,*] Dongpeng Wang[2] King Hung Chiu[2] Ngai Wong[1]
[1] Department of Electrical and Electronic Engineering, The University of Hong Kong, Hong Kong
[2]United Microelectronics Centre (Hong Kong) Limited, Hong Kong Science Park, N.T., Hong Kong
Email Address: {linrui, jieran, nwong}@eee.hku.hk, {khchiu}@umechk.com

Abstract

Recent results have revealed an interesting observation in a trained convolutional neural network (CNN), namely, the rank of a feature map channel matrix remains surprisingly constant despite the input images. This has led to an effective rank-based channel pruning algorithm [23], yet the constant rank phenomenon remains mysterious and unexplained. This work aims at demystifying and interpreting such rank behavior from a frequency-domain perspective, which as a bonus suggests an extremely efficient Fast Fourier Transform (FFT)-based metric for measuring channel importance without explicitly computing its rank. We achieve remarkable CNN channel pruning based on this analytically sound and computationally efficient metric, and adopt it for repetitive pruning to demonstrate robustness via our scheme named ***E****nergy-****Z****oned* ***C****hannels for* ***R****obust* ***O****utput* ***P****runing (EZCrop), which shows consistently better results than other state-of-the-art channel pruning methods. The codes and Appendix are publicly available at: https://github.com/ruilin0212/EZCrop.*

1. Introduction

Convolutional neural networks (CNNs) are among the most popular models for deep learning which have achieved breakthroughs in vision applications including classification [41], object detection [28], semantic segmentation [34], etc. However, deeper and larger CNNs render them challenging to deploy on edge devices with constrained hardware resources. This dilemma has motivated the research on compressing CNNs for lower storage and faster inference. Numerous neural network compression techniques have been proposed, including optimized implementation [16], quantization [2, 5], network pruning [10], low-rank decomposition [25, 17] and knowledge distillation [12, 33], etc. Among the categories mentioned above, this work focuses on channel pruning, a subclass of structured network pruning.

*RL, JR contributed equally to this work.

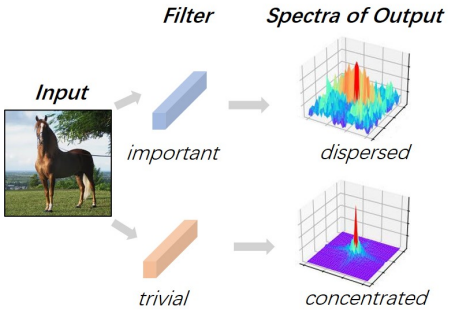

Figure 1. The spectral of output is obtained by applying 2D-FFT and $\mathrm{fftshift}(\cdot)$ on the output in the spatial domain. (Upper) The output of an important filter in the frequency domain has a dispersed energy distribution. (Lower) The output of a trivial filter has a concentrated energy distribution in the frequency domain.

Due to its outstanding performance in model size reduction and network acceleration, network pruning has been a popular compression strategy. Generally, it can be divided into two subclasses: weight pruning and filter pruning. Weight pruning [7, 8] tends to eliminate small weights in the kernel tensors, leading to a sparse architecture with fewer nonzero connections. However, it is not easy to achieve inference acceleration via the existing efficient BLAS libraries due to the unstructured sparsity pattern, whose utilization requires specialized software or hardware. On the other hand, channel pruning [20, 29, 23] removes entire filters in the kernel tensor. Compared with weight pruning, eliminating entire filters results in structured sparsity and generic speedup irrespective of the software/hardware, which promotes the usefulness of filter pruning.

The key challenge in channel pruning is to evaluate the importance of filters and select trivial or unimportant candidates for pruning. For instance, in [35], the authors propose a novel criterion based on Taylor expansion to evaluate the filters' importance and discard the less important ones. In [31], the authors use 2D discrete cosine transform (DCT) on the filters to exploit the spatial correlations in frequency domain and prune filters containing less low-frequency com-

ponents dynamically. In [10] "smaller-norm-less-important" filter pruning is proposed, which removes filters with smaller norms. Recently, Lin *et al.* [23] observe that the average rank of feature maps generated by a fixed filter remains surprisingly constant irrespective of the input images. They then propose a scheme called *HRank* to prune channels with small corresponding ranks and thereby less information content.

Noticeably, most existing pruning schemes operate in the spatial domain, with only a few of them utilizing information from the frequency or spectral domain. Inspired by the work of Sedghi *et al.* [39] that studies the singular values in a convolution layer from a Fast Fourier Transformation (FFT) viewpoint, we take FFT as a tool for deciphering the constant rank behavior [23], and develop a filter pruning scheme with a novel, analytical, energy-based explanation. Our key insight is that the energy distributions of feature maps generated by a fixed filter can reflect the filter's importance (Figure 1). Alongside this observation, we propose a novel frequency-domain channel pruning approach called Energy-Zoned Channel for robust output pruning (EZCrop) , which entails a natural, accurate, and more robust selection of important filters at a lower computational cost than the singular value decomposition (SVD) needed for evaluating ranks in HRank [23].

In order to make a comprehensive comparison versus HRank [23] and other state-of-the-art algorithms [20, 13, 29, 32, 45, 15, 27, 10, 44, 19, 3, 42, 22, 30, 6, 24, 21] , we benchmark EZCrop for image classification tasks on CIFAR-10 [18] and ImageNet [4], using VGGNet [40], ResNet [9], and DenseNet [14]. The experimental results demonstrate the *consistent superiority* of EZCrop in both model reduction and acceleration. Moreover, we conduct repetitive pruning with HRank and EZCrop, which further illustrates the high robustness of EZCrop. Our main contributions are:

- We analytically bridge the rank-based channel importance metric in the spatial domain to an energy perspective in the frequency domain, and for the first time explain the interesting constant-rank phenomenon in a channel matrix.
- We propose a computationally efficient FFT-based metric for channel importance, which reduces the computational complexity from $\mathcal{O}(n^3)$ to $\mathcal{O}(n^2 \log n)$.
- The proposed EZCrop algorithm for channel pruning is simple, intuitive and robust. It outperforms other state-of-the-art pruning schemes and consistently delivers good performance, which is not a result of normal variations as confirmed through extensive experiments.

2. Related Work

2.1. Filter Pruning

In contrast to weight pruning that results in unstructured sparsity, channel or filter pruning belongs to the class of structured pruning and can readily leverage the efficient BLAS library without specific software or hardware requirements. Generally, there are two categories of CNN filter pruning schemes, namely, by utilizing post-training properties of the CNN itself or by adaptive pruning during training. The first category employs algebraic metrics. For example, in [20], the authors use l_1-norm of the filters or the corresponding feature maps to evaluate the importance, where smaller norms mean less informative channels. In HRank [23], filters with small corresponding average ranks calculated by a batch of feature map slices are pruned. For the second category, adaptive pruning approaches make decisions through retraining with a specialized loss function taking pruning into account. For example, Liu *et al.* [29] impose sparsity-induced regularization on scaling factors in the batch normalization layers, where channels with zero-valued scaling factors are pruned. By reformulating the batch normalization layer, the authors of [46] propose a new parameter called saliency, and only channels with saliency beyond a threshold during training can remain.

2.2. Application of Frequency Information in the Convolutional Neural Networks

In recent years, the properties of CNNs in the frequency domain have attracted increasing attention due to the acceleration of CONV calculations and the additional information arising from the frequency domain. In [38], the authors propose spectral pooling that preserves more information and complex-coefficient spectral parameterization of filters that improves training convergence. Pratt *et al.* [37] propose a frequency-domain CNN, whose training can be done entirely in the frequency-domain without alternating between spatial and spectral domains. In [39], by using FFT properties, the authors design a regularizer to constrain the range of CONV kernels' singular values during model training, which improves the model performance. Including [31], there are other works using spectral information for network compression. For example, in [1], the model is compressed in a frequency-sensitive fashion which preserves parameters of low-frequency components better. The filters are treated as images in [43], where the authors compress the model by discarding the low-energy coefficients of filters in the frequency domain.

Summarizing over the above works on filter pruning and frequency-domain CNN design, we notice the research on filter pruning through exploiting frequency-domain information is insufficient. The method in [31], though doing dynamic pruning in the frequency domain, belongs to unstructured and adaptive pruning, and suffers from machine and labor costs. Therefore, further exploring the use of spectral information for filter pruning is desirable.

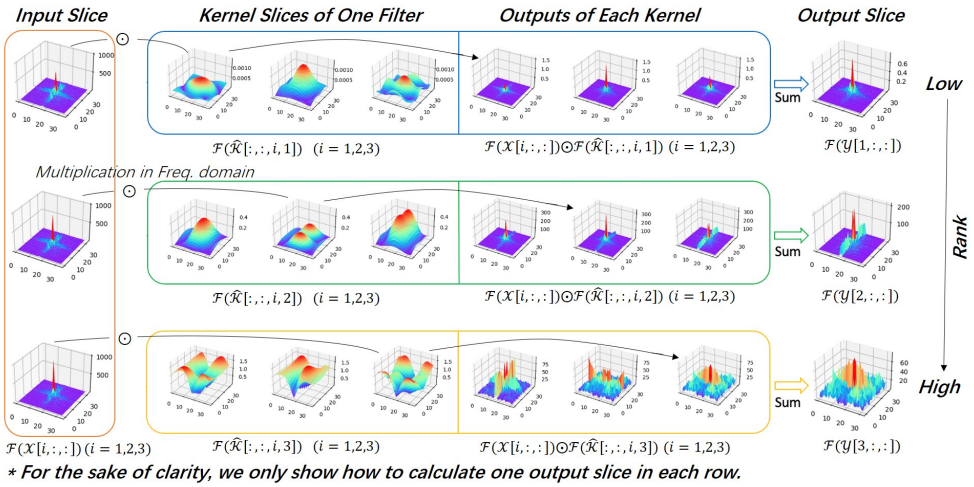

Figure 2. The first column shows the three input channels in the frequency domain with center shifting, namely, $\text{fftshift}(\mathcal{F}(\boldsymbol{\mathcal{X}}[i,:,:]))$ for $i = 1, 2, 3$. On the left of the second column are expended kernel slices $\text{fftshift}(\mathcal{F}(\hat{\boldsymbol{\mathcal{K}}}[:,:,i,j]))$, where $i = 1, 2, 3$ and j is fixed but different in three rows. The corresponding outputs slices $\text{fftshift}(\mathcal{F}(\hat{\boldsymbol{\mathcal{K}}}[:,:,i,j]) \odot \mathcal{F}(\boldsymbol{\mathcal{X}}[i,:,:]))$ are shown on the right part, and $\mathcal{F}(\boldsymbol{\mathcal{Y}}[j,:,:])$ are the sum of three element-wise product, respectively. The selected filters generate feature maps with increasing average ranks from top to bottom. Note the spectral energy distribution of the feature map becomes more dispersed for higher rank spatial filters.

3. Constant Ranks in Feature Map: From a Perspective in the Frequency Domain

The idea proposed in [39] by using FFT to constrain the singular values of kernels inspires us to use FFT to relate the rank-based metric in the spatial domain to an energy perspective in the frequency domain.

3.1. Notation

For a trained CNN model, we use $\boldsymbol{\mathcal{Y}}_i \in \mathbb{R}^{B\times T_i\times H_i\times W_i}$ to denote the output[1] of the i-th CONV layer ($i = 1, 2, \cdots, N$), where B is the batch size, T_i is the number of channels, H_i and W_i are the height and width, respectively. We denote the kernels of the i-th CONV layer as $\boldsymbol{\mathcal{K}}_i \in \mathbb{R}^{D_i\times D_i\times S_i\times T_i}$, where D_i represents the kernel size, S_i and T_i are the number of input and output channels. We use $\boldsymbol{Y}_i^j[b,:,:] \in \mathbb{R}^{H_i\times W_i}$ to denote the feature map generated by the b-th sample in the batch passing through the j-th filter in the i-th CONV layer. The set $I_i = \{I_i^1, I_i^2, \cdots, I_i^{T_i}\}$ is used to record the importance of filters in the i-th CONV layer. Every element in the set I_i records the index of one filter. We use $\mathcal{F}(\cdot)$ and $\mathcal{F}^{-1}(\cdot)$ to denote FFT and inverse FFT. The notation $\text{fftshift}(\cdot)$ stands for the operation swapping four quadrants to center the DC component, $\text{abs}(\cdot)$ means element-wise magnitude, and $\text{ceil}(\cdot)$ means rounding up.

3.2. Convolution in the Frequency Domain

It is well-known that convolution in the time/spatial domain is equivalent to multiplication in the frequency domain. In [36], the author illustrates how to transform a general 2D convolution into a Fourier-based one. In brief, there are three steps: 1) for the given convolution kernel and input data, we apply 2D FFT on them respectively, 2) then we do point-wise multiplication between the newly obtained kernel and input, 3) lastly, we perform inverse 2D FFT on the multiplied result. Theorem 1 formalizes the convolution process in the frequency domain, and the detailed proof is provided in Appendix 1.

Theorem 1 *For a single 3-D input $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{S\times H\times W}$ and a given filter $\boldsymbol{\mathcal{K}} \in \mathbb{R}^{D\times D\times S\times T}$, their convolution result $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{T\times H\times W}$ can be formalized as:*

$$\boldsymbol{\mathcal{Y}}[j,:,:] = \sum_{i=1}^{S} \mathcal{F}^{-1}(\mathcal{F}(\hat{\boldsymbol{\mathcal{K}}}[:,:,i,j]) \odot \mathcal{F}(\boldsymbol{\mathcal{X}}[i,:,:])), \quad (1)$$

where $\odot$ stands for the point-wise multiplication (also called Hadamard product), and $\hat{\boldsymbol{\mathcal{K}}} \in \mathbb{R}^{H\times W\times S\times T}$ is the expanded filter, whose slices $\hat{\boldsymbol{\mathcal{K}}}[:,:,i,j]$ are the torus form of the corresponding slices $\boldsymbol{\mathcal{K}}[:,:,i,j]$ in the original filter.

By writing Eq. 1 alternatively as

$$\mathcal{F}(\boldsymbol{\mathcal{Y}}[j,:,:]) = \sum_{i=1}^{S} \mathcal{F}(\hat{\boldsymbol{\mathcal{K}}}[:,:,i,j]) \odot \mathcal{F}(\boldsymbol{\mathcal{X}}[i,:,:]), \quad (2)$$

the input, filter, and the corresponding output are all in the frequency domain. The equivalence between spatial convolution and spectral point-wise product establishes the credibility of using frequency-domain information. Taking a sample image from CIFAR-10 and three filters from the first CONV layer in a pre-trained VGGNet, Figure 2 visualizes convolution in the frequency domain. More visualization results are available in Appendix 2.1.

[1]The output selected here is after the operations like ReLU, batch normalization, and max pooling, etc. In other words, the output we extract for the i-th CONV layer is the input for the $(i+1)$-th CONV layer.

3.3. Energy Zoning

According to the rightmost column row of Figure 2, the concentration of low-frequency components of the generated feature map is inversely proportional to the average rank of the feature maps. Specifically, the average rank of the feature maps generated by the filter selected in the first row is the smallest, its low-frequency components of the feature map are the most concentrated. In the third row, the selected filter generates feature maps with high average ranks, and it is obvious that the frequency components of its feature map are much more dispersed. This phenomenon shows that a filter generating feature maps with high average ranks in the spatial domain is expected to generate feature maps with dispersed frequency components. Figure 3 uses a toy example to conceptually depict a low-rank channel matrix and a high-rank one. We use zero-valued and nonzero-valued elements to represent high-frequency and low-frequency components, respectively. It is obvious that the right matrix with a high rank has more dispersed nonzero elements[2].

To precisely quantify, we define a new metric called *energy zone* due to its underlying nature of summing spectral components related to power. The energy zone here refers to an area obtained by expanding outward from the DC center. Due to the conjugate symmetry after applying $\mathcal{F}(\cdot)$ and $\mathrm{fftshift}(\cdot)$ on the feature map, we define the energy zone with a **symmetrical** shape. For **best visualization**, we choose **circle** as the energy-zone shape, and choose a feature map with a large spatial size.

Figure 4 shows the results taking the first CONV in VGGNet having 64 filters as an example, the energy in each circle takes up 70% of the total feature map. The details of how to draw the circles are provided in Appendix 2.1. The 64 subfigures are arranged in an ascending order of the rank-based HRank scheme in [23] from left to right and top to bottom. In general, we can see that the radius of the circle increases as the average rank of the feature maps increases, which shows the high consistency between the rank-based scheme in the spatial domain and the dispersion of the feature maps' low-frequency components in the frequency domain. However, the circle radii are not monotonically increasing, which means there are still subtle differences in the evaluation of certain filters based on information in the spatial and frequency domains. We remark that analyzing the low-frequency components' concentration brings in **higher resolution** in channels' importance evaluation since every element in $H_i \times W_i$ is considered. In contrast, the rank of each feature map slice is only an integer in $[0, min(H_i, W_i)]$. This is the reason why HRank and energy metric may produce subtle discrepancy in their channel importance measure. More discussions on the rank-based and energy-based metrics are in Appendix 3.

[2]We remark that the rank of a channel matrix is *invariant* in the spatial and spectral domains as 2D FFT is a bijective linear mapping.

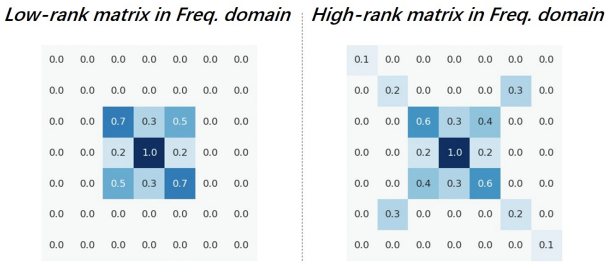

Figure 3. Connection between ranks and energy dispersion. When transforming the matrix to the frequency domain, the $\mathrm{fftshift}(\cdot)$ operation is employed, which is essentially a block permutation and is rank-preserving. Here it is apparent the left matrix is only of rank 3 while the right is full-rank, and such spectral ranks also translate to spatial ranks due to rank-invariant domain transforms.

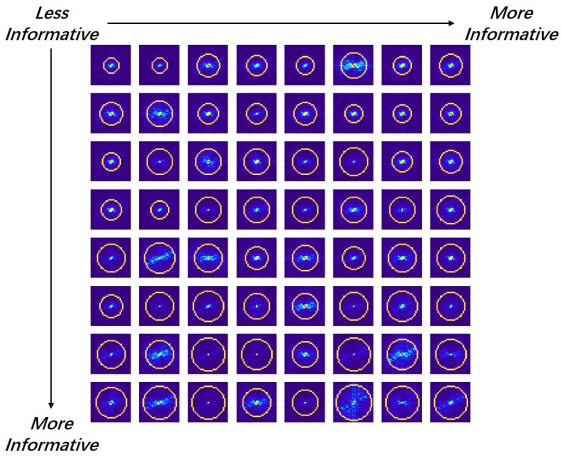

Figure 4. The outputs are generated by 64 filters from the first CONV layer of a pretrained VGGNet. The 64 subfigures are arranged in an ascending order of the rank-based HRank scheme from left to right and top to bottom. We determine the radii of the circles to measure and visualize the concentration of the low-frequency components of the feature map in the frequency domain.

To further benchmark the difference in filter importance evaluation, we propose a pruning metric based on the energy zone in Section 4 and evaluate it through various experiments in Section 5.

4. EZCrop

Recall from Section 3.3 that the radius can be used to evaluate the concentration of the feature map's frequency components. In particular, filters with a large average corresponding radius are treated as important ones. However, this method **loses resolution** when H_i and W_i of the feature maps $\boldsymbol{\mathcal{Y}}_i$ are **small**. For example, it is difficult to determine a proper radius when the feature map slice is of size 4×4 or smaller, such as 2×2. Subsequently, a simple metric based on the concept of energy zone regardless of the feature maps' size is desired. Here, we set the shape of an energy

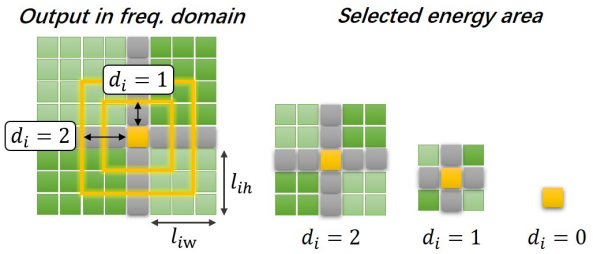

Figure 5. An example to illustrate how to find the square center, and determine the selected energy area.

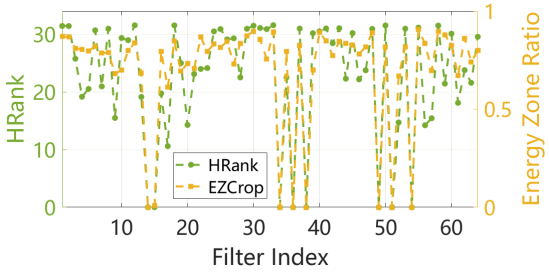

Figure 6. The two curves indicate the importance of the 64 filters using HRank and our energy-based metrics (Eq. 5) in the spatial and spectral domains, respectively. The dashed lines joining the points are only for the ease of visualization wherein the points are discrete in nature.

zone to be **square** and evaluate the importance of filters by four steps:1) find the square center, 2) decide the size of the square, 3) calculate the energy-zone ratios, and 4) sort the filters according to the ratios. These steps are detailed below and the pseudocode can be found in Appendix 4.2.

Step 1: Find the Square Center. For a given feature map $\boldsymbol{\mathcal{Y}}_i$, as long as H_i and W_i are known, we can determine the coordinate of the DC center by

$$x_i = \begin{cases} \frac{H_i}{2} + 1, & H_i \text{ is even} \\ \frac{H_i+1}{2}, & H_i \text{ is odd} \end{cases}, \tag{3a}$$

$$y_i = \begin{cases} \frac{W_i}{2} + 1, & W_i \text{ is even} \\ \frac{W_i+1}{2}, & W_i \text{ is odd} \end{cases}, \tag{3b}$$

regardless of the parity of H_i and W_i or whether H_i and W_i are equal. Figure 5 visualize the square center when H_i and W_i are equal and are both even, and the schematic diagrams for other scenarios are available in Appendix 4.1.

Step 2: Decide the Expanding Distance. The size of a energy-zoned square is constrained by limiting the distance away from the center, accordingly:

$$\begin{aligned} l_{ih} &= H_i - x_i,\ l_{iw} = W_i - y_i, \\ d_i &= \begin{cases} 0, & x_i - 1 = 0 \text{ or } y_i - 1 = 0 \\ \text{ceil}(\beta \cdot \min(l_{ih}, l_{iw})), & \text{else} \end{cases}, \end{aligned} \tag{4}$$

where $\beta \in (0, 1)$ is a hyper-parameter to limit the area. It is worth noting that, $d_i = 0$ means only the DC component itself is contained in the energy-zoned square. Figure 5 shows the examples when $d_i = 0, 1, 2$.

Step 3: Calculate the Energy Zone Ratios. Different from Section 3.3, we determine the size of the energy zone first instead of the ratio of the sum of the elements in the energy zone. In this case, we evaluate the concentration of feature map's low-frequency components by the proportion of the sum of elements outside the energy-zoned square:

$$\eta_i^j = \frac{1}{B} \cdot \left(1 - \sum_{b=1}^{B} \frac{S(d_i[b])}{S(\boldsymbol{E}_i^j[b,:,:])}\right), \tag{5}$$

where $S(d_i[b])$ and $S(\boldsymbol{E}_i^j[b,:,:])$ denote the sum of elements in the energy-zoned square with d_i as the extended length and the sum of elements in the whole square, respectively, for the b-th sample in a batch, and η_i^j is the energy zone ratio of j-th filter in the i-th CONV layer, where larger value means more important the filter is. In Figure 6, we compare the results of the evaluation of the filters' importance through HRank [23] and EZCrop. Consistent with the conclusion of Section 3.3, the two metrics generally track each other, especially for particularly important or unimportant channels/filters. For filters between the two poles, the evaluation results are slightly different. In Section 5, we use sufficient experiments to show EZCrop indeed provides a better scrutiny of important filters.

Step 4: Sort the Filters. After finishing the above steps, for each CONV layer, we can obtain a set of energy zone ratios $\eta_i = \{\eta_i^1, \eta_i^2, \cdots, \eta_i^{T_i}\}$. The set $I_i = \{I_i^1, I_i^2, \cdots, I_i^{T_i}\}$ records the channel index of the elements in η_i after sorting them in a descending order. For example, if η_i^{12} ranks first in η_i, then $I_i^1 = 12$. Figure 7 shows an example how we sort the filters according to the energy ratios.

Complexity Analysis. For a given feature map slice $\boldsymbol{Y}_i^j[b,:,:] \in \mathbb{R}^{H_i \times W_i}$, we compare the complexity of HRank and EZCrop to generate the pruning metrics in spatial and frequency domains, respectively. HRank [23] employs the function $\text{torch.matrix\_rank}(\cdot)$ in PyTorch, which uses SVD to compute the matrix rank at a complexity of $\mathcal{O}(\max(H_i, W_i)^3)$. For EZCrop, applying 2D FFT to transform $\boldsymbol{Y}_i^j[b,:,:]$ to the frequency domain is the most time-consuming step, which we implemented by $\text{np.fft.fft2}(\cdot)$ in NumPy with a complexity of $\mathcal{O}(H_i W_i \log(H_i W_i))$. For a simpler expression, we rewrite it as $\mathcal{O}(n^2 \log n)$ which indicates the high efficiency of EZCrop versus that of HRank.

5. Experimental Results

We conduct extensive experiments on image classification tasks, using CIFAR-10 [18] and ImageNet [4] datasets, to demonstrate the superiority of the proposed EZCrop channel pruning scheme. Popular CNN models are used, including VGGNet [40], ResNet [9], and DenseNet [14]. We present the performance of state-of-the-art algorithms for a comprehensive overview, and we mainly compare EZCrop against its closest scheme HRank which uses channel matrix rank in the spatial domain to measure channel importance.

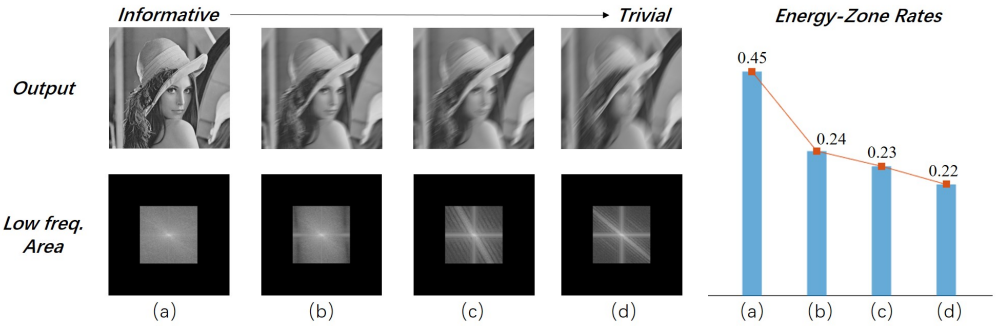

Figure 7. By assuming the pictures as the outputs generated by four filters, we give an example to show intuitively how energy-zone ratios reflect the importance of the corresponding filters using the frequency information. (Left) The first row contains four feature map slices, the quality of which decreases from left to right. The second row shows the corresponding selected energy areas. It is worth noting that the concentration of the energy distribution increases as the feature maps become less informative. (Right) The more information the feature map contains, the larger the energy-zone ratio, and the more important the corresponding filter is.

The pruning results shown in Tables 2 to 5 are directly acquired from the corresponding papers or their official GitHub repositories. We obtained the result of multi-pass HRank in Table 6 ourselves by generalizing the single-pass scheme, since the authors of [23] did not report the results of applying HRank repetitively. For simplicity, we fix $\beta = 0.25$ in Eq. 4 in all experiments when generating the energy zone ratios, more training details can be found in Appendix 5. All experiments are run on a computer equipped with four NVIDIA GeForce GTX1080Ti Graphics Cards, each with 11GB frame buffer.

Time Comparison. In Table 1, we further show the actual runtimes of EZCrop and HRank to generate the required pruning metrics for those CIFAR-10/ImageNet networks on one GTX1080Ti GPU. Apparently, the time required by EZCrop to generate the pruning metric is much shorter in all models. Noticeably, the larger the dataset, the more pronounced the efficiency of EZCrop is.

Dataset	Model	HRrank [23]	**EZCrop** (↓)
CIFAR-10	VGGNet	1505.54s	**356.94**s (76.29%)
	ResNet-56	1247.51s	**381.97**s (69.38%)
	DenseNet-40	473.17s	**171.50**s (63.76%)
ImageNet	ResNet-50	7.96h	**3.45**h (56.66%)

Table 1. Runtimes of HRank and EZCrop to compute pruning metrics on CIFAR-10 / ImageNet for different models.

5.1. CIFAR-10

VGGNet. The VGG-16 used in the experiments is a variation of the original VGGNet for CIFAR-10 dataset taken from [20]. In Table 2, the upper part is the performance of state-of-the-art pruning approaches excluding HRank, while the lower part focuses on the comparison between HRank and EZCrop (such convention also applies to subsequent tables). First, we compare EZCrop with the upper-part approaches. Compared with FPGM, though both can reach the accuracy of around 94%, EZCrop has higher computational efficiency with more significant FLOPs reduction (58.1% vs. 35.9%). When compared with L1, SSS, Zhao *et al.* , GAL, and PScratch, EZCrop excels in all aspects wherein it obtains the fastest and most compact model with the best performance. Furthermore, when compared with HRank under the same settings, namely, same FLOPs and Params reductions, EZCrop outperforms HRank in both cases (94.01% vs. 93.73% / 93.70% vs. 93.56%). This reveals that though HRank can evaluate the filters well using the information in the spatial domain, EZCrop offers more accurate evaluation benefiting from the additional frequency information.

Model	Top-1%	FLOPs (↓)	Params (↓)
VGGNet	93.96	313.73M(0.0%)	14.98M(0.0%)
L1 [20]	93.40	206.00M(34.3%)	5.40M(64.0%)
SSS [15]	93.02	183.13M(41.6%)	3.93M(73.8%)
Zhao *et al.* [46]	93.18	190.00M(39.1%)	3.92M(73.3%)
GAL-0.05 [27]	92.03	189.49M(39.6%)	3.36M(77.6%)
GAL-0.1 [27]	90.78	171.89M(45.2%)	2.67M(82.2%)
FPGM [10]	94.00	201.10M(35.9%)	–
PScratch [44]	93.63	156.86M(50.0%)	–
HRank [23]	93.73	131.17M(58.1%)	2.76M(81.6%)
EZCrop	**94.01**	**131.17**M(58.1%)	**2.76**M(81.6%)
HRank [23]	93.56	104.78M(66.6%)	2.50M(83.3%)
EZCrop	**93.70**	**104.78**M(66.6%)	**2.50**M(83.3%)

Table 2. Pruning results of VGGNet on CIFAR-10.

ResNet-56. Table 3 shows the results of different pruning schemes on ResNet-56. When the FLOPs reductions are around 30%, EZCrop outperforms L1, LeGR, and HRank. Compared with L1 and HRank (first row in the lower part), EZCrop discards more parameters (34.1%) than L1 (14.1%) and HRank (22.3%), while reaches the highest accuracy (94.18%). As for LeGR, EZCrop squeezes out more accuracy (94.18%) than it (94.08%) under the same FLOPs reduction (30%). Compared with He *et al.*, FPGM and PScratch, when their FLOPs reductions are both around 50%, EZCrop has the highest accuracy of 93.80%. When compared with GAL and NISP, EZCrop excels in all aspects. EZCrop also outperforms DHP, as the gap between their

Model	Top-1%	FLOPs (↓)	Params (↓)
ResNet-56	93.26	125.49M(0.0%)	0.85M(0.0%)
L1 [20]	93.06	90.90M(27.6%)	0.73M(14.1%)
He *et al.* [11]	90.80	62.00M(50.6%)	–
NISP [45]	93.01	81.00M(35.5%)	0.49M(42.4%)
GAL-0.6 [27]	92.98	78.30M(37.6%)	0.75M(11.8%)
GAL-0.8 [27]	90.36	49.44M(60.2%)	0.29M(65.9%)
FPGM [10]	93.49	59.44M(52.6%)	–
PScratch [44]	93.05	62.75M(50.0%)	–
LeGR [3]	94.10	87.84M(30.0%)	–
SCOP [42]	93.64	55.22M(56.0%)	0.37M(56.3%)
DHP [22]	92.94	49.07M(60.9%)	0.35M(58.9%)
Hinge [24]	93.72	66.40M(47.1%)	–
HRank [23]	93.85	90.35M(28.0%)	0.66M(22.3%)
EZCrop	**94.18**	**87.84**M(30.0%)	**0.56**M(34.1%)
HRank [23]	93.57	65.94M(47.4%)	0.48M(42.8%)
EZCrop	**93.80**	**65.94**M(47.4%)	**0.48**M(42.8%)
HRank [23]	92.32	34.78M(74.1%)	0.24M(70.0%)
EZCrop	**92.52**	**34.78**M(74.1%)	**0.24**M(70.0%)

Table 3. Pruning results of ResNet-56 on CIFAR-10.

Model	Top-1%	FLOPs (↓)	Params (↓)
DenseNet-40	94.82	282.00M(0.0%)	1.04M(0.0%)
Liu *et al.*-40% [29]	**94.81**	190.00M(32.8%)	0.66M(36.5%)
GAL-0.01 [27]	94.29	182.92M(35.3%)	0.67M(35.6%)
GAL-0.05 [27]	93.53	128.11M(54.7%)	0.45M(56.7%)
Zhao *et al.* [46]	93.16	156.00M(44.8%)	0.42M(59.7%)
HRank [23]	94.51	173.39M(38.5%)	0.62M(40.1%)
EZCrop	**94.72**	**173.39**M(38.5%)	**0.62**M(40.1%)
HRank [23]	93.66	113.08M(59.9%)	0.39M(61.9%)
EZCrop	**93.76**	**113.08**M(59.9%)	**0.39**M(61.9%)

Table 4. Pruning results of DenseNet-40 on CIFAR-10.

accuracy is only around 0.4% at FLOPs reduction $> 14\%$. Besides, EZCrop and SCOP show comparable performance under similar FLOPs reduction. When the FLOPs reduction is around 47%, EZCrop outperforms Hinge. For the comparative experiments between EZCrop and HRank, EZCrop can achieve higher accuracy under all settings. This shows EZCrop not only outperforms HRank on networks with plain structures but also residual blocks.

DenseNet-40. The results of DenseNet-40 are given in Table 4. Compared with Liu *et al.*, EZCrop achieves higher compression rate (Params: 0.62M vs. 0.66M) and faster speed (FLOPs: 173.39M vs. 190.00M) with only 0.09% lower accuracy. This shows EZCrop is more promising to achieve better performance when highly compact models are required. Compared with GAL-0.01, EZCrop has advantages in all aspects. As for GAL-0.05 and Zhao *et al.*, when the Params reductions are all around 60%, EZCrop obtains the best performance (Top-1%: 93.76%) with the highest FLOPs reduction (59.9%). In the two comparative experiments with HRank, EZCrop achieves higher Top-1 accuracy in both cases. This verifies that EZCrop beats HRank also on networks with inception modules.

5.2. ImageNet

ResNet-50. The results on ImageNet, which is a more complicated dataset than CIFAR-10, are shown in Table 5. When the number of parameters is around 15M, EZCrop (75.68%) has higher accuracy than SSS-26 (71.82%) and GAL-1 (69.31%). Though GAL-1 can reduce the FLOPs more than EZCrop (FLOPs: 1.58B vs. 2.26B), its accuracy drop of 6.37% is drastic. When the number of parameters is around 10M, EZCrop outperforms ThiNet-50 and GAL-1-joint with obvious advantages on the performance (74.33% vs. 66.42% / 69.31%). Compared with MetaPruning, EZCrop has better performance (75.68% vs. 75.40%) with higher FLOPs reduction. For **DMCP** and **EagleEye**, though they obtain more compact models with higher accuracy, they require dramatically more GPU resources. DMCP needs 8$\sim$15h for training to find the possible structure, using 16 32GB 1080Ti cards. EagleEye comprises three stages, namely, strategy proposal, candidate selection and fine-tuning. Excluding the fine-tuning stage, EagleEye takes more than 1 GPU day (an Nvidia 2080Ti card) to search for acceptable candidates with no guarantee that the best one can be selected. As depicted in Table 1, EZCrop costs only 3.45h (a single 11GB 1080Ti card) to obtain the pruning metric. In other words, EagleEye requires $> 10\times$ more GPU resources than EZCrop, which leads to a mere 0.7% accuracy improvement after fine-tuning. For other upper part approaches, EZCrop excels in all aspects. For the comparative experiments with HRank, EZCrop boosts the accuracy in all cases. This indicates that EZCrop can deal with large and complicated datasets as well.

Model	Top-1%	Top-5%	FLOPs	Params
ResNet-50 [32]	76.15	92.87	4.09B	25.50M
He *et al.* [11]	72.30	90.80	2.73B	–
ThiNet-50 [32]	68.42	88.30	1.10B	8.66M
SSS-26 [15]	71.82	90.79	2.33B	15.60M
SSS-32 [15]	74.18	91.91	2.82B	18.60M
GDP-0.5 [26]	69.58	90.14	1.57B	–
GDP-0.6 [26]	71.19	90.71	1.88B	–
GAL-0.5 [27]	71.95	90.94	2.33B	21.20M
GAL-1 [27]	69.88	89.75	1.58B	14.67M
GAL-0.5-joint [27]	71.80	90.82	1.84B	19.31M
GAL-1-joint [27]	69.31	89.12	1.11B	10.21M
FPGM [10]	75.91	92.63	2.36B	–
MetaPruning [30]	75.40	–	2.29B	–
DMCP [6]	76.20	–	2.20B	–
EagleEye [19]	76.40	92.89	2.00B	–
ABCPrunner-80% [21]	73.86	91.69	1.89B	11.75M
HRank [23]	75.56	92.63	2.26B	15.09M
EZCrop	**75.68**	**92.70**	2.26B	15.09M
HRank [23]	74.19	91.94	1.52B	11.05M
EZCrop	**74.33**	**92.00**	1.52B	11.05M

Table 5. Pruning Results of ResNet-50 on ImageNet.

5.3. Robustness Under Repetitive Pruning

We further test HRank and EZCrop in a multi-pass manner, i.e., pruning models repetitively. Repetitive pruning requires multiple evaluations of channels' importance, which amplifies the difference between the two schemes. We conduct the experiments using ResNet-56 on CIFAR-10, and keep adopting HRank and EZCrop to get a more compact model without a fixed compression rate, then compare their

performance. For fairness, all settings of HRank and EZCrop for every single-pass are the same. Table 6 shows that when getting more compact models, the gap of EZCrop over HRank is enlarged after each single-pass. When we reduce the #Params from 0.85M to 0.63M, the gap between the accuracy of HRank (93.76%) and EZCrop (93.95%) is 0.19%. After the second pass, the gap is enlarged to 0.27% with 0.17M more parameters discarded. In the final pass, the gap is further expanded to 0.6% while more than 50% parameters are reduced. This obviously shows that EZCrop enjoys higher robustness under repetitive pruning, and is able to evaluate the filters more accurately in each single-pass.

#Passes (#epochs)	FLOPs	Params	HRank [23]	**EZCrop**
1 (300)	$90.86M$	$0.63M$	93.76%	**93.95%**
2 (300)	$66.25M$	$0.46M$	93.15%	**93.42%**
3 (300)	$36.03M$	$0.22M$	91.58%	**92.18%**

Table 6. Repetitive pruning of ResNet-56 on CIFAR-10.

5.4. Standard Deviation (STD) Analysis

To further show the effectiveness and robustness of EZCrop while dispelling the worries that the improvements compared with HRank are due to standard variations, we conduct the first comparative experiments in Tables 2-4 for 20 times and compute the average and STD of the accuracy, respectively. The results in Table 7 prove that the improvements are not accidental, and EZCrop actually makes more accurate and robust evaluations of the importance of filters.

Mean Acc. (%) / STD	VGGNet	ResNet-56	DenseNet-40
EZCrop	**93.98/0.073**	**93.99/0.097**	**94.63/0.066**
HRank [23]	93.75/0.140	93.76/0.168	94.26/0.165

Table 7. The mean and STD of the accuracy by repeating each selected experimental setting for 20 times.

5.5. Ablation Study

Excluding the pruning metrics, the compression rate and training settings for VGGNet / ResNet-56 / DenseNet-40 in this section are the same as the first comparative experiments in Tables 2-4, respectively.

5.5.1 Effectiveness of Energy-zone Rate

To show that EZCrop makes reliable filter importance evaluation, we compare the results by pruning the models: 1) using EZCrop, 2) randomly, 3) using inverse EZCrop (filters with high energy-zone ratio will be removed). For each network in Figure 8, we observe that EZCrop outperforms random pruning, while the inverse EZCrop has the worst performance. The gaps in the accuracy confirm the effectiveness of our energy-zone metric in the frequency domain.

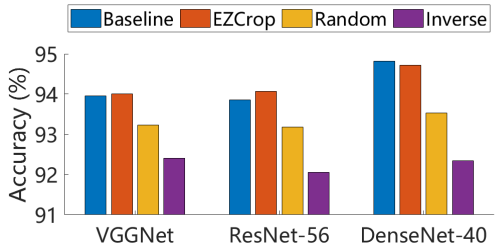

Figure 8. Performances of different pruning approaches when pruning VGGNet / ResNet-56 / DenseNet-40 on CIFAR-10.

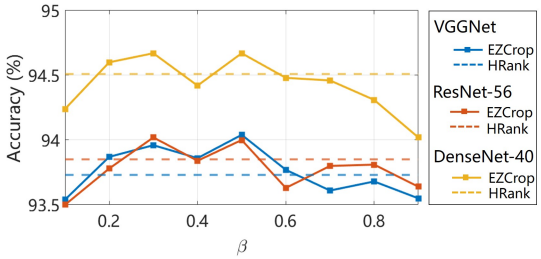

Figure 9. Relations between accuracy and β, which decides the expanding distance. The larger β, the larger the energy zone.

5.5.2 Appropriate Expending Distance

There is a hyper-parameter β in EZCrop when we decide the expanding distance. In this paper, we set $\beta = 0.25$ for all experiments for the sake of simplicity. However, this general setting may not be an optimal setting for all. In Figure 9, we explore the relations between the accuracy and β. It is worth noting that the three curves have similar trends, the curve shows an "M" shape when β is between 0.2 and 0.6. Based on this observation, we suggest to manually set β around 0.3 or 0.5, namely, two peaks of the curve.

6. Conclusion

This work has connected the previously mysterious constant-rank phenomenon in a CNN feature map channel to a novel, analytical view in the frequency domain. Via a spectral perspective, an efficient FFT-based energy-zone metric has been proposed for evaluating the importance of a channel and its corresponding filter. This leads to a channel pruning scheme named EZCrop (energy-zoned channels for robust output pruning) which readily outperforms existing state-of-the-art in terms of compression rates, runtimes and FLOPs. EZCrop also constitutes a robust way for repetitive channel pruning to stretch the limit of CNN compression, while maintaining high output accuracy.

Acknowledgement

This work is supported in part by the General Research Fund (GRF) projects 17209721 & 17206020, and in part by the in-kind support of United Microelectronics Centre (Hong Kong) Limited.

References

[1] Wenlin Chen, James Wilson, Stephen Tyree, Kilian Q Weinberger, and Yixin Chen. Compressing convolutional neural networks in the frequency domain. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1475–1484, 2016.

[2] Jian Cheng, Jiaxiang Wu, Cong Leng, Yuhang Wang, and Qinghao Hu. Quantized cnn: A unified approach to accelerate and compress convolutional networks. *IEEE transactions on neural networks and learning systems*, 29(10):4730–4743, 2017.

[3] Ting-Wu Chin, Ruizhou Ding, Cha Zhang, and Diana Marculescu. Legr: Filter pruning via learned global ranking. 2019.

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[5] Ruizhou Ding, Ting-Wu Chin, Zeye Liu, and Diana Marculescu. Regularizing activation distribution for training binarized deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11408–11417, 2019.

[6] Shaopeng Guo, Yujie Wang, Quanquan Li, and Junjie Yan. Dmcp: Differentiable markov channel pruning for neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1539–1547, 2020.

[7] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[8] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pages 1135–1143, 2015.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] Yang He, Ping Liu, Ziwei Wang, Zhilan Hu, and Yi Yang. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4340–4349, 2019.

[11] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1389–1397, 2017.

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

[13] Hengyuan Hu, Rui Peng, Yu-Wing Tai, and Chi-Keung Tang. Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. *arXiv preprint arXiv:1607.03250*, 2016.

[14] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[15] Zehao Huang and Naiyan Wang. Data-driven sparse structure selection for deep neural networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 304–320, 2018.

[16] Chunhui Jiang, Guiying Li, Chao Qian, and Ke Tang. Efficient dnn neuron pruning by minimizing layer-wise nonlinear reconstruction error. In *IJCAI*, volume 2018, pages 2–2, 2018.

[17] Taehyeon Kim, Jieun Lee, and Yoonsik Choe. Bayesian optimization-based global optimal rank selection for compression of convolutional neural networks. *IEEE Access*, 8:17605–17618, 2020.

[18] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[19] Bailin Li, Bowen Wu, Jiang Su, and Guangrun Wang. Eagleeye: Fast sub-net evaluation for efficient neural network pruning. In *European Conference on Computer Vision*, pages 639–654. Springer, 2020.

[20] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*, 2016.

[21] Yawei Li, Shuhang Gu, Christoph Mayer, Luc Van Gool, and Radu Timofte. Group sparsity: The hinge between filter pruning and decomposition for network compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8018–8027, 2020.

[22] Yawei Li, Shuhang Gu, Kai Zhang, Luc Van Gool, and Radu Timofte. Dhp: Differentiable meta pruning via hypernetworks. *arXiv preprint arXiv:2003.13683*, 2020.

[23] Mingbao Lin, Rongrong Ji, Yan Wang, Yichen Zhang, Baochang Zhang, Yonghong Tian, and Ling Shao. Hrank: Filter pruning using high-rank feature map. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1529–1538, 2020.

[24] Mingbao Lin, Rongrong Ji, Yuxin Zhang, Baochang Zhang, Yongjian Wu, and Yonghong Tian. Channel

pruning via automatic structure search. *arXiv preprint arXiv:2001.08565*, 2020.

[25] Shaohui Lin, Rongrong Ji, Chao Chen, Dacheng Tao, and Jiebo Luo. Holistic cnn compression via low-rank decomposition with knowledge transfer. *IEEE transactions on pattern analysis and machine intelligence*, 41(12):2889–2905, 2018.

[26] Shaohui Lin, Rongrong Ji, Yuchao Li, Yongjian Wu, Feiyue Huang, and Baochang Zhang. Accelerating convolutional networks via global & dynamic filter pruning. In *IJCAI*, pages 2425–2432, 2018.

[27] Shaohui Lin, Rongrong Ji, Chenqian Yan, Baochang Zhang, Liujuan Cao, Qixiang Ye, Feiyue Huang, and David Doermann. Towards optimal structured cnn pruning via generative adversarial learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2799, 2019.

[28] Yudong Liu, Yongtao Wang, Siwei Wang, TingTing Liang, Qijie Zhao, Zhi Tang, and Haibin Ling. Cbnet: A novel composite backbone network architecture for object detection. In *AAAI*, pages 11653–11660, 2020.

[29] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2736–2744, 2017.

[30] Zechun Liu, Haoyuan Mu, Xiangyu Zhang, Zichao Guo, Xin Yang, Kwang-Ting Cheng, and Jian Sun. Metapruning: Meta learning for automatic neural network channel pruning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3296–3305, 2019.

[31] Zhenhua Liu, Jizheng Xu, Xiulian Peng, and Ruiqin Xiong. Frequency-domain dynamic pruning for convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1043–1053, 2018.

[32] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE international conference on computer vision*, pages 5058–5066, 2017.

[33] Ping Luo, Zhenyao Zhu, Ziwei Liu, Xiaogang Wang, Xiaoou Tang, et al. Face model compression by distilling knowledge from neurons. In *AAAI*, pages 3560–3566, 2016.

[34] Yawei Luo, Liang Zheng, Tao Guan, Junqing Yu, and Yi Yang. Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2507–2516, 2019.

[35] Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*, 2016.

[36] Victor Podlozhnyuk. Fft-based 2d convolution. *NVIDIA white paper*, 32, 2007.

[37] Harry Pratt, Bryan Williams, Frans Coenen, and Yalin Zheng. Fcnn: Fourier convolutional neural networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 786–798. Springer, 2017.

[38] Oren Rippel, Jasper Snoek, and Ryan P Adams. Spectral representations for convolutional neural networks. In *Advances in neural information processing systems*, pages 2449–2457, 2015.

[39] Hanie Sedghi, Vineet Gupta, and Philip M Long. The singular values of convolutional layers. *arXiv preprint arXiv:1805.10408*, 2018.

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

[41] Yanan Sun, Bing Xue, Mengjie Zhang, Gary G Yen, and Jiancheng Lv. Automatically designing cnn architectures using the genetic algorithm for image classification. *IEEE transactions on cybernetics*, 50(9):3840–3854, 2020.

[42] Yehui Tang, Yunhe Wang, Yixing Xu, Dacheng Tao, Chunjing Xu, Chao Xu, and Chang Xu. Scop: Scientific control for reliable neural network pruning. *arXiv preprint arXiv:2010.10732*, 2020.

[43] Yunhe Wang, Chang Xu, Shan You, Dacheng Tao, and Chao Xu. Cnnpack: Packing convolutional neural networks in the frequency domain. In *Advances in neural information processing systems*, pages 253–261, 2016.

[44] Yulong Wang, Xiaolu Zhang, Lingxi Xie, Jun Zhou, Hang Su, Bo Zhang, and Xiaolin Hu. Pruning from scratch. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12273–12280, 2020.

[45] Ruichi Yu, Ang Li, Chun-Fu Chen, Jui-Hsin Lai, Vlad I Morariu, Xintong Han, Mingfei Gao, Ching-Yung Lin, and Larry S Davis. Nisp: Pruning networks using neuron importance score propagation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9194–9203, 2018.

[46] Chenglong Zhao, Bingbing Ni, Jian Zhang, Qiwei Zhao, Wenjun Zhang, and Qi Tian. Variational convolutional neural network pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2780–2789, 2019.