Feature Variance Ratio-Guided Channel Pruning for Deep Convolutional Network Acceleration

Junjie He[0000−0002−1009−945X], Bohua Chen[0000−0003−4467−1060], Yinzhang Ding[0000−0001−9375−9400], and Dongxiao Li*[0000−0002−5619−0419]

Zhejiang University, Hangzhou 310027, China
{he_junjie, chenbohua, dingyzh, lidx}@zju.edu.cn

Abstract. Most existing channel pruning approaches utilize the magnitude of network parameters to guide the pruning process. However, these methods suffer from some limitations in modern networks, where the magnitude of parameters can vary independently of the importance of corresponding channels. To recognize redundancies more accurately and therefore, accelerate networks better, we propose a novel channel pruning criterion based on the Pearson correlation coefficient. The criterion preserves the features that are essentially informative to the given task and avoids the influence of useless parameter scales. Based on this criterion, we further establish our channel pruning framework named Feature Variance Ratio-guided Channel Pruning (FVRCP). FVRCP prunes channels globally with little human intervention. Moreover, it can automatically find important layers in the network. Extensive numerical experiments on CIFAR-10 and ImageNet with widely varying architectures present state-of-the-art performance of our method.

1 Introduction

Deep convolutional neural networks (CNNs) have achieved state-of-the-art performance in various computer vision tasks [1–4]. One essential foundation of such great success lies in the deep and wide architectures, which are always accompanied by the expensive computational costs. As a result, deploying these models on resource-constrained devices (e.g., smartphones and IoT systems) becomes extremely difficult. To address this issue, various model acceleration methods [5–9] have been proposed to improve the computational efficiency of CNNs. Among them, channel pruning [10–15] becomes a prevalent one due to its hardware-friendly implementation and surprising ability to reduce a large amount of computation overhead without compromising model performance.

Channel pruning aims to remove redundant channels in the convolutional layers. Existing practices in this field for identifying unimportant channels mainly resort to the magnitude of parameters, e.g., the norm of channel weights [16, 17], the norm of filters [10, 13], and the absolute value of scaling factors of batch normalization (BN) layers [12, 14]. These methods hypothesize that the channels

* Corresponding author

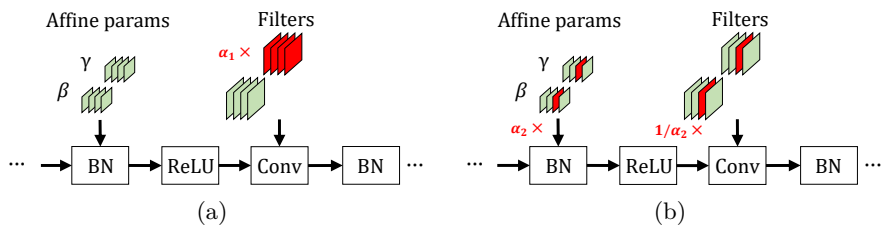


Fig. 1. Illustration of two equivalent transformations for a typical network: (a) scale a filter by α_1 ($\alpha_1>0$); (b) scale a pair of affine parameters of BN by α_2 ($\alpha_2>0$) and inversely scale the corresponding channel weights in the next convolutional layer. γ and β are scaling and shifting factors, respectively. These transformations change the magnitude of parameters but do not change the function of *any* channels.

with small magnitude parameters contribute little to the network and then are less important. However, it can be shown that the parameter magnitude cannot faithfully characterize the channel importance in modern networks. For the typically used convolutional network with BN and rectified linear unit (ReLU)[1] as shown in Fig. 1, there are two equivalent transformations [19, 20] by which we can change the magnitudes of network parameters arbitrarily without changing the information flow (and therefore the importance of each channel) in the network: 1) scale the filters in an intermediate layer by a positive factor, since BN normalizes convolutional output and the scaling effect of filters is canceled; 2) scale the affine parameters of BN by a positive factor and simultaneously inversely scale the corresponding channel weights in the next layer, as ReLU is positively homogeneous which satisfies $\text{ReLU}(\alpha x) = \alpha\text{ReLU}(x)$ for all $\alpha>0$. A robust channel importance metric should be invariant to these transformations while the conventional magnitude-based ones do not. This suggests that despite good acceleration ratios achieved, existing pruning methods are still suboptimal. The magnitudes of parameters are less relevant to the identification of channel importance. Useful features may be falsely discarded with the approaches based on them, which severely impair model performance.

In this paper, with the goal of recognizing redundancies more accurately and therefore, accelerating networks better, we propose a novel channel pruning criterion based on the Pearson correlation coefficient. Specifically, we exploit the Pearson correlation coefficient to assess the information loss of convolutional output feature maps resulted from pruning. The feature variance ratio is then constructed as an importance metric to guide the pruning process. Different from the conventional magnitude-based metrics, the new metric *avoids* the influence of the scale of the convolutional filters and the scale of the affine parameters of BN layers. It is interpretable from the feature-correlation perspective and easy to calculate. With proposed metric, we can directly prune channels for a pre-trained network in a single-shot (i.e., globally prune once) with little fine-tuning while still preserving its high performance. After that, we further

[1] ReLU can be replaced with other positively homogeneous functions like PReLU [18].

establish our channel pruning framework, named Feature Variance Ratio-guided Channel Pruning (FVRCP). In contrast to prior works [11, 21, 22] that require handcrafted layer-wise pruning ratios, FVRCP prunes channels globally with little human intervention. It automatically finds the important layers for the network, which inspires us to design better architectures.

The numerical experiments on CIFAR-10 and ImageNet with widely varying architectures present state-of-the-art performance of FVRCP. For example, on the large-scale ImageNet dataset, when pruning 40% FLOPs of PreResNet-50, FVRCP improves the original PreResNet model by 0.05% in top-1 accuracy; when pruning 43% FLOPs of MobileNets, FVRCP causes zero accuracy drop, exceeding the uniform baseline [23] by 2.2%.

The major contributions of this paper are summarized as follows: First, we propose a novel interpretable channel importance metric based on the Pearson correlation coefficient. The new metric avoids the influence of parameter scales and identifies the channels that are essentially informative to the given task. Second, we propose FVRCP framework to prune channels for CNNs globally with little human intervention. FVRCP automatically finds compact structures in the network and reduces the overwhelming computational burden.

2 Related Work

Weight Pruning. Convolutional neural network acceleration has been extensively studied in recent years. Weight pruning [24, 9, 25–28] tries to find and remove unimportant connections in the network. Early work [29] in the 1990s uses the Hessian of loss function to determine the importance of connections. Recently, Han *et al.* [9, 25] propose an iterative method that prunes connections with small weights. Dong *et al.* [28] prune the parameters of each layer based on a layer-wise error function. Though these methods achieve high compression ratios, they result in irregular networks, of which the speedup can only be achieved in specialized software or hardware that supports sparse matrix operation.

Channel Pruning. To be free of customized platforms and extra operations, many approaches [10, 21, 11, 13, 12, 14, 30, 31, 16, 32] directly prune regular channels for the network. Li *et al.* [13] prune channels based on the ℓ_1-norm of filters. He *et al.* [10] select unimportant channels with an ℓ_2-norm criterion. Liu *et al.* [12] leverage the scaling factors in batch normalization layer to remove insignificant channels. Liu and Sun [31] employ an evolutionary algorithm with a meta network to search for the best channel pruning strategies. In these works, the magnitudes of network parameters are always utilized to establish their pruning frameworks [10, 21, 13, 12, 14]. However, as shown in Fig. 1, the information transmitted by the magnitude of parameters about the importance of network channels can be extremely limited. Besides, most approaches [10, 21, 11, 13, 22, 15] require human experts to design layer-wise pruning ratios. Determining these layer-wise pruning ratios not only requires specialized knowledge but also greatly reduces the search space of pruning under which we cannot achieve the optimal compression ratio.

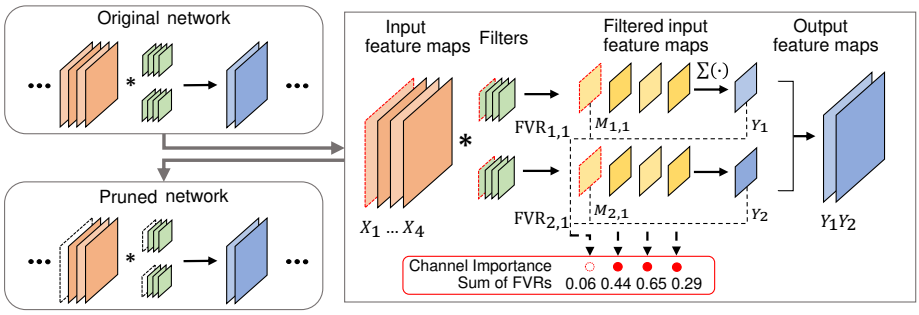


Fig. 2. Feature variance ratio-guided channel pruning. The channel with small sum of FVRs (*red dotted elements in the right side*) will be pruned.

Other Methods. Apart from pruning, there are many other excellent works for CNN accelerations, such as knowledge distillation [8, 33], quantization [34, 7, 35, 36], and low-rank decomposition [5, 6]. All these approaches are orthogonal to our work, and one can combine them to accelerate networks further.

3 Methodology

We first introduce some notations that will be used throughout this paper. Let $X \in \mathbb{R}^{C \times H_{in} \times W_{in}}$ and $Y \in \mathbb{R}^{N \times H_{out} \times W_{out}}$ be a standard convolutional input and output feature maps, respectively. C and N denote the number of input and output channels. Moreover, let X_i be the i-th feature map in X, Y_j be the j-th feature map in Y, and $K_{j,i}$ be the $D_h \times D_w$ kernel corresponding to X_i and Y_j.

Batch normalization [19] enables faster training and better generalization of deep CNNs and now is becoming a standard component in deep learning. We focus on the batch normalized networks in this paper, but it should be noted that our method can be extended to the general networks (see supplementary). Recall that BN is normally inserted immediately after convolution, which normalizes convolutional outputs by subtracting the mean and dividing the standard deviation, and then rescale and re-shift them.

3.1 Construction of Feature Variance Ratio

To facilitate the investigation of the importance of each channel, we first consider the convolution with a single filter $K_{j,:}$. The corresponding output is:

$$Y_j = \sum_{i=1}^{C} M_{j,i} = \sum_{i=1}^{C} K_{j,i} * X_i, \tag{1}$$

where $*$ denotes 2D convolution operation and $M_{j,i} = K_{j,i} * X_i$ denotes the filtered input feature map.

Channel pruning tries to pick up some channels to discard. If the pruning of a channel does not hurt the inherent information encoded in the output Y_j, then this channel is redundant and can be removed. Instead of the Mean Square Error (MSE) that prior works [22, 11] used, we employ the Pearson correlation coefficient to measure the information loss caused by pruning. This comes from the fact that the scale and bias of output feature maps are normalized by the following BN. The inherent information of Y_j is substantially encoded in its normalized version, or more specifically, the direction of the vector represented by the normalized version in high-dimensional feature space. Pearson correlation coefficient, which characterizes the linear correlation between two variables or equivalently the geometrical angle between two zero-mean vectors, is more precise to describe this loss.

Suppose that we prune the i_0-th channel from the input tensor X. Then the output feature map becomes:

$$Y'_j = \sum_{i=1, i\neq i_0}^{C} M_{j,i}. \tag{2}$$

The Pearson correlation coefficient between Y_j and Y'_j is defined as:

$$r_{j,i_0} = \frac{\sum_{p,q}\left(y'_{j,p,q} - \bar{y'}_j\right)\left(y_{j,p,q} - \bar{y}_j\right)}{\sqrt{\sum_{p,q}\left(y'_{j,p,q} - \bar{y'}_j\right)^2}\sqrt{\sum_{p,q}\left(y_{j,p,q} - \bar{y}_j\right)^2}}, \tag{3}$$

where $y_{j,p,q}$, $y'_{j,p,q}$ represent the (p,q)-th elements of Y_j and Y'_j respectively, and $\bar{y}_j$, $\bar{y'}_j$ are their corresponding average values. Larger absolute value of the Pearson correlation coefficient indicates more linear association between Y_j and Y'_j and thus less information loss caused by the pruning of i_0-th channel. When the Pearson correlation coefficient has the value $r_{j,i_0} = \pm 1$, there is no information loss. In other words, the distance between 1 and r^2_{j,i_0} can be exploited to determine the information richness of i_0-th input channel (and thus its importance).

However, the most widely recognized disadvantage of the Pearson correlation coefficient is that it is computationally intensive. Especially here we should compute it for the thousands of channels in the network. We hope to have a metric that not only reflects the channel importance effectively but also is computationally efficient. To this end, we propose the Feature Variance Ratio (FVR) indicator. Specifically, it is the variance ratio of the filtered feature map of X_{i_0}, i.e., M_{j,i_0}, to the output feature map Y_j, and can be calculated as:

$$\mathrm{FVR}_{j,i_0} = \frac{\sigma^2_{Mj,i_0}}{\sigma^2_{Y_j}} = \frac{\frac{1}{H_{out}W_{out}}\sum_{p,q}\left(m_{j,i_0,p,q} - \bar{m}_{j,i_0}\right)^2}{\frac{1}{H_{out}W_{out}}\sum_{p,q}\left(y_{j,p,q} - \bar{y}_j\right)^2}, \tag{4}$$

where $m_{j,i_0,p,q}$ represents the (p,q)-th element of M_{j,i_0} and $\bar{m}_{j,i_0}$ is the average value. Intuitively, FVR_{j,i_0} implies the strength of X_{i_0} to Y_j. When FVR_{j,i_0} is small, X_{i_0} becomes an offset component of Y_j, and pruning it will not destroy the inherent feature structure of output feature map. More importantly, it can be shown that FVR is highly correlated with Pearson correlation coefficient.

Proposition 1 *Let x, y be two n-dimensional data vectors with elements $\{x_i\}$ and $\{y_i\}$, respectively. Assume their Pearson correlation coefficient is r_{xy}, i.e.,*

$$r_{xy} = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2}\sqrt{\sum_i (y_i - \bar{y})^2}}, \tag{5}$$

where $\bar{x} = \sum_i x_i/n, \bar{y} = \sum_i y_i/n$. Let $\varepsilon_i = y_i - x_i, i = 1, 2, ..., n$, be the residuals of data elements, forming a vector ε. Suppose that the variances of ε and y are σ_ε^2, σ_y^2 respectively, and $\sigma_y^2 > 0$, $\sigma_\varepsilon^2 \neq \sigma_y^2$. Then we have:

$$0 \leq 1 - r_{xy}^2 \leq \frac{\sigma_\varepsilon^2/\sigma_y^2}{(1 - \sigma_\varepsilon/\sigma_y)^2}. \tag{6}$$

We provide the proof in the supplementary material. By Proposition 1 and the fact that $Y_j = Y_j' + M_{j,i_0}$, we obtain:

$$0 \leq 1 - r_{j,i_0}^2 \leq \frac{\mathrm{FVR}_{j,i_0}}{(1 - \sqrt{\mathrm{FVR}_{j,i_0}})^2}. \tag{7}$$

Eq. 7 reveals the relationship between FVR_{j,i_0} and r_{j,i_0}^2. The right side of this equation is monotonically increasing with increasing FVR_{j,i_0} in $[0, 1]$, and tends to zero as $\mathrm{FVR}_{j,i_0} \to 0$. This implies that smaller FVR_{j,i_0} corresponds to the smaller $1 - r_{j,i_0}^2$, and in the limit case we have:

$$\lim_{\mathrm{FVR}_{j,i_0} \to 0} 1 - r_{j,i_0}^2 = 0. \tag{8}$$

From the above, we conclude that FVR_{j,i_0} is an effective alternative metric for measuring importance of X_{i_0} to Y_j. The smaller FVR_{j,i_0} is, the less information loss caused by pruning X_{i_0}, and thus the less important the i_0-th channel is.

3.2 Channel Importance Based on FVR

The importance metric FVR mentioned in Section 3.1 is constructed on a single output feature map. In fact, there is more than one filter in a convolutional layer. An input feature map engages in the formation of multiple output feature maps. To completely measure the importance of a channel in the network, we take the sum of its corresponding FVRs across all output feature maps, as Fig. 2 illustrated:

$$\mathrm{SFVR}_{i_0} = \sum_{j=1}^{N} \mathrm{FVR}_{j,i_0} = \sum_{j=1}^{N} \frac{\sigma_{M_{j,i_0}}^2}{\sigma_{Y_j}^2}. \tag{9}$$

We define SFVR (omit i_0 for clarity) as the channel importance and prune the channels that have small ones. Indeed, the essence of this pruning philosophy is similar to that of Principal Component Analysis (PCA) for dimensionality reduction. PCA discards the principal components that have small variances in

the original data to reduce dimensions, since these components possess little energy and disposing of them will not lead to great loss. Here we also discard the components (input channels) that have small variances because they are less informative to the given task. Dividing by the variances of output feature maps can be regarded as a normalization technique for output dimensions.

Despite the different numbers of output channels, which affects the summation, the channel importance defined in Eq. 9 is comparable across all layers. Consider a situation where the filtered input feature maps $M_{j,i}, i = 1, \ldots, C$, are orthogonal each other with the same variance $\sigma^2_{M_j}$, and the number of input channels is equal to that of output channels, i.e., $C = N$. Then we have $\mathrm{SFVR}_{i_0} = \sum_{j=1}^{N} \sigma^2_{M_j}/\sigma^2_{Y_j} = \sum_{j=1}^{N} 1/C = 1$, which is a constant independent of layers. It illustrates that if the channels of all layers are utilized fully and equally, the pruning criterion will not prefer to prune any layers, which is consistent with our intuition. Moreover, if the filtered input feature maps $M_{j,i}, i = 1, \ldots, C$, are identical to each other with the variance $\sigma^2_{M_j}$, then $\mathrm{SFVR}_{i_0} = \sum_{j=1}^{N} \sigma^2_{M_j}/\sigma^2_{Y_j} = \sum_{j=1}^{N} 1/C^2 = 1/C$, which implies that the layer with more channels will be pruned first. It is also natural since in this case the layers that have more channels are more redundant. All these suggest that SFVR can be compared across different layers, and it will automatically find important layers in the network. More details can be seen in Section 4.4.

Linear Transformation Invariance. A key mathematical property of SFVR is that it is invariant under separate changes in bias and scale in output feature maps. That is, for any scalar a_j ($a_j \neq 0$) and b_j, $\widehat{Y}_j = a_j Y_j + b_j$, we have:

$$\sum_{j=1}^{N} \frac{\sigma^2_{\widehat{M}_{j,i_0}}}{\sigma^2_{\widehat{Y}_j}} = \sum_{j=1}^{N} \frac{a_j^2 \sigma^2_{M_{j,i_0}}}{a_j^2 \sigma^2_{Y_j}} = \sum_{j=1}^{N} \frac{\sigma^2_{M_{j,i_0}}}{\sigma^2_{Y_j}}, \tag{10}$$

where $\widehat{M}_{j,i_0} = a_j M_{j,i_0} + d_{j,i_0}$ ($\sum_{i=1}^{C} d_{j,i} = b_j$) is the transformed feature map of M_{j,i_0}. The property implies that SFVR is insensitive to the scale and bias of output feature maps, which is not surprising since we have considered that the influence of scale and bias of convolutional output feature maps will be canceled by the following BN transform and they do not encode any information.

Network Equivalent Transformation Invariance. As mentioned earlier, due to the positively homogenous property of ReLU and normalization process of BN, there exist two equivalent transformations by which we can change the magnitude of network parameters without altering the function of corresponding channels. A robust channel importance metric should be invariant to these changes while the conventional magnitude-based ones do not. In contrast, it can be easily verified that our proposed SFVR is not affected by these transforms (detailed in the supplementary). It is more effective in identifying redundant channels in modern networks.

3.3 Implementation via Moving Average Statistics

The proposed channel importance metric SFVR is composed of two parts, namely, $\sigma^2_{Y_j}$ and $\sigma^2_{M_{j,i_0}}$. To apply in practice, we need to estimate them.

Algorithm 1 Algorithm Description of FVRCP

Initialize: planned pruning ratio $r = 0$
gradual pruning schedule $\mathcal{T}_{\mathcal{N},\mathcal{R}}(\cdot)$
for *epoch* $= 1$ to $\mathcal{N}$ **do**
  Update parameters and moving average statistics
  Compute SFVR for each channel as Eq. 12
  Let $r \leftarrow \mathcal{T}_{\mathcal{N},\mathcal{R}}(epoch)$
  while (pruned FLOPs ratio $< r$) **do**
    Remove the channel with smallest SFVR
  end while
end for
Fine-tune the pruned model until it converges

Given the fact that BN normalizes convolutional outputs, we can directly leverage the statistical information provided by BN to estimate $\sigma^2_{Y_j}$ without any additional computation. Instead of batch sample statistics, we employ moving average statistics of BN for the stability and reliability of estimate. However, since the mainstream implementation of convolution is *im2col*, which transforms whole convolution into matrix multiplication [37, 38], we cannot directly compute $\sigma^2_{M_{j,i_0}}$. Although we can decompose the regular convolution into several fine-grained operations such as depthwise convolution with summation, it will greatly affect the speed of forward computation of the model on modern computing devices like GPUs. To address this issue, we consider the following relaxation:

$$\begin{aligned}\sigma^2_{M_{j,i_0}} &= \frac{1}{H_{out}W_{out}} \sum_{p,q} \left(m_{j,i_0,p,q} - \bar{m}_{j,i_0}\right)^2 \\ &= \frac{1}{H_{out}W_{out}} \sum_{p,q} \left| \left(R_{i_0,p,q} - \bar{R}_{i_0}\right) * K_{j,i_0} \right|^2 \\ &\leq \|K_{j,i_0}\|_F^2 \, \frac{1}{H_{out}W_{out}} \sum_{p,q} \left\| R_{i_0,p,q} - \bar{R}_{i_0} \right\|_F^2 ,\end{aligned} \tag{11}$$

where $\|\cdot\|_F$ denotes the Frobenius norm, $R_{i_0,p,q}$ denotes the receptive field of $m_{j,i_0,p,q}$ on X_{i_0}, and $\bar{R}_{i_0} = \sum_{p,q} R_{i_0,p,q}/(H_{out}W_{out})$. The right side of Eq. 11 is an upper bound for $\sigma^2_{M_{j,i_0}}$. It contains only two terms. One is the Frobenius norm of kernel weights, and the other is the statistic of input feature map X_{i_0}. Both of them can be conveniently computed without any modification to the network architecture. We use this bound to approximate $\sigma^2_{M_{j,i_0}}$. Same with $\sigma^2_{Y_j}$, to stabilize the estimation, we retain the moving average statistics of input feature maps in each training iteration and employ them when pruning.

By the approximation of $\sigma^2_{M_{j,i_0}}$, the channel importance then we calculate for the i_0-th input channel becomes:

$$\mathrm{SFVR}^*_{i_0} = \frac{1}{H_{out}W_{out}} \sum_{p,q} \left\| R_{i_0,p,q} - \bar{R}_{i_0} \right\|_F^2 \sum_{j=1}^{N} \frac{\|K_{j,i_0}\|_F^2}{\sigma^2_{Y_j}}. \tag{12}$$

Table 1. Comparison of pruning results on CIFAR-10. "Global" indicates whether the method is a global channel pruning algorithm.

Model	Method	Global	Baseline Acc.	Pruned Acc.	Acc. ↓	FLOPs	FLOPs ↓ (%)
ResNet-56	Li *et al.* [13]	No	93.04	93.06	-0.02	90.9M	27.6
	NISP [30]	Yes	-	-	0.03	-	43.6
	CP [11]	No	92.8	91.8	1.0	-	50.0
	FVRCP-50	Yes	**93.41**	**93.64**	**-0.23**	**62.6M**	**50.1**
ResNet-110	Li *et al.* [13]	No	93.53	93.30	0.23	155M	38.6
	NISP [30]	Yes	-	-	0.18	-	43.8
	GAL [40]	Yes	93.50	92.74	0.76	130.2M	48.5
	FVRCP-70	Yes	**94.06**	**93.86**	**0.20**	**75.8M**	**70.0**
DenseNet-40	Liu *et al.* [12]	Yes	93.89	94.35	-0.46	120M	57.6
	C-SGD [32]	No	93.81	94.56	-0.75	113.0M	60.1
	GAL [40]	Yes	94.81	93.23	1.58	80.9M	71.4
	FVRCP-60	Yes	**94.83**	**94.60**	**0.23**	**113.0M**	**60.1**
	FVRCP-75	Yes	**94.83**	**93.91**	**0.92**	**70.6M**	**75.0**

From Eq. 11, we see that SFVR* puts more emphasis on the channels that are filtered by the larger kernels, which is acceptable since larger kernels have stronger learning abilities.

3.4 Channel Pruning Framework

With a well-trained convolutional network, our proposed Feature Variance Ratio-guided Channel Pruning (FVRCP) procedures are illustrated in Algorithm 1.

We employ the gradual pruning technique in [39]. During pruning, the FLOPs pruning ratio is increased quadratically from zero to the preset target ratio ($\mathcal{R}$ in Algorithm 1), and at the end of each epoch, we globally select the channels that have small importance to remove until achieving the planned pruning ratio at that epoch. The gradual pruning technique helps smooth the pruning process and prevents the algorithm from degradation. The final fine-tuning is also an important technique to recover pruned accuracy.

4 Experiments

In this section, we evaluate our channel importance metric and channel pruning framework on CIFAR-10 [41] and ImageNet [42] with several popular architectures. All the experiments are implemented using TensorFlow [43] on NVIDIA TITAN V GPUs.

4.1 Experimental Settings

On CIFAR-10, we conduct gradual pruning for 40 epochs with a mini-batch size of 128 and a fixed learning rate of 0.01. The standard data augmentation is

Table 2. Comparison of pruning results on ImageNet.

Model	Method	Global	Baseline Top1	Baseline Top5	Pruned Top1	Pruned Top5	FLOPs	FLOPs ↓ (%)
ResNet-18	SFP [10]	No	70.28	89.63	67.10	87.78	1.06B	41.8
	FPGM [21]	No	70.28	89.63	68.34	88.53	1.06B	41.8
	FVRCP-42	Yes	**70.23**	**89.36**	**68.88**	**88.39**	**1.05B**	**42.0**
ResNet-50	ThiNet [22]	No	72.88	91.14	72.04	90.67	2.59B	36.8
	C-SGD [32]	No	75.33	92.56	75.27	92.46	2.59B	36.8
	SFP [10]	No	76.15	92.87	74.61	92.06	2.38B	41.8
	HRank [44]	No	76.15	92.87	74.98	92.33	2.30B	43.8
	CP [11]	No	-	92.2	-	90.8	-	50.0
	FPGM [21]	No	76.15	92.87	74.83	92.32	1.90B	53.5
	Hinge [47]	Yes	-	-	74.70	-	1.90B	53.5
	FVRCP-40	Yes	**76.09**	**92.90**	**76.04**	**92.92**	**2.45B**	**40.0**
	FVRCP-50	Yes	**76.09**	**92.90**	**75.42**	**92.52**	**2.04B**	**50.0**
PreResNet-50	FVRCP-40	Yes	**76.01**	**92.86**	**76.06**	**92.82**	**2.45B**	**40.0**
	FVRCP-50	Yes	**76.01**	**92.86**	**75.49**	**92.47**	**2.04B**	**50.0**
	FVRCP-60	Yes	**76.01**	**92.86**	**74.93**	**92.33**	**1.63B**	**60.0**

Table 3. Comparison of pruned MobileNet V1 on ImageNet. The latency is tested on TITAN V GPU and Intel Xeon E3-1230 CPU with a batch size of 32.

Model	FLOPs	Top1 Acc.	Latency GPU	Latency CPU
Baseline [23]	569M	70.6%	1.23ms	17.27ms
0.75× MobileNet V1 [23]	325M	68.4%	0.95ms	12.23ms
FVRCP-43	**324M**	**70.6%**	**0.91ms**	**11.18ms**
NetAdapt [48]	284M	69.1%	-	-
FVRCP-50	**284M**	**69.8%**	**0.88ms**	**10.28ms**
0.5× MobileNet V1 [23]	149M	63.7%	0.71ms	7.48ms
FVRCP-74	**149M**	**65.5%**	**0.69ms**	**6.35ms**

adopted including padding and random cropping. After that, we fine-tune the model for 40 epochs with a learning rate of 0.001. On ImageNet, we conduct gradual pruning for 20 epochs. The mini-batch size is 256. The learning rate during pruning is 0.01. The fine-tuning is for 20 epochs with an initial learning rate of 0.001 and decayed by 0.1 at 10 epochs. All networks are trained using stochastic gradient descent (SGD) with Nesterov momentum 0.9.

4.2 Comparison with State-of-the-Art Methods

Table 1 shows our channel pruning results on CIFAR-10 dataset. As we see, FVRCP method achieves state-of-the-art performance. With 50.0% FLOPs reduction for ResNet-56, CP [11] causes 1.0% loss in accuracy while FVRCP improves 0.23%. With almost zero performance loss on ResNet-110, FVRCP

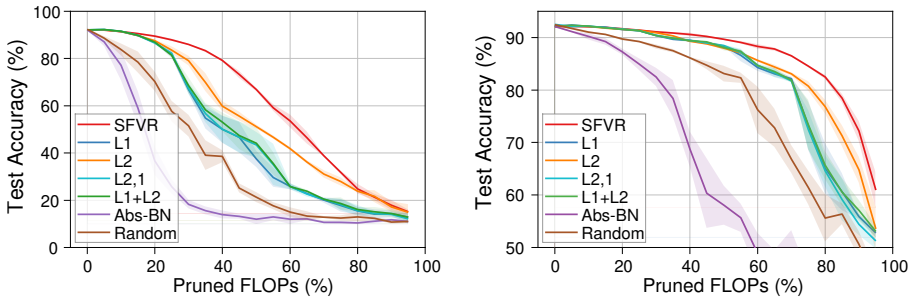


Fig. 3. Single-shot pruning without fine-tuning (*left*) and with 10 fine-tuning epochs (*right*) on pre-activation ResNet-20 on CIFAR-10. All experiments are repeated 5 times with different random seeds. ± standard derivation is reported with shaded region.

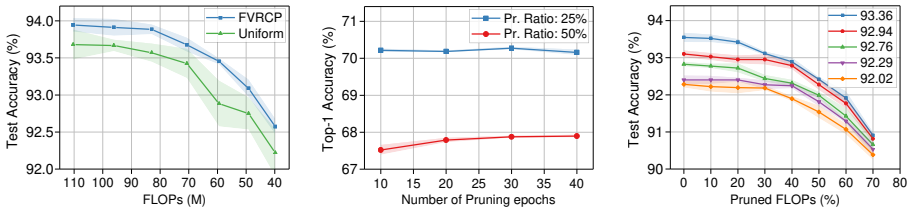
achieves a 70.0% FLOPs reduction, much higher than the 38.6% reduction by Li *et al.* [13] who use ℓ_1-norm of filters to guide the pruning process.

We further evaluate FVRCP on the large-scale ImageNet dataset, as shown in Table 2. Again, FVRCP presents outstanding performance. On ResNet-18, FVRCP achieves the same theoretical speedup with FPGM [21] and SFP [10], but its top-1 accuracy surpasses FPGM by more than 0.5% and significantly exceeds 67.1% obtained by SFP which is heuristically based on the ℓ_2-norm of filters. For ResNet-50, FVRCP reduces 50% FLOPs with only 0.38% top-5 accuracy drop, outperforming CP [11] by 1.02%. Moreover, note that almost all compared methods cannot prune channels globally for the network. These approaches require human experts to carefully design layer-wise pruning ratios (e.g., sensitive analysis [13]), which is laborious in practice.

MobileNets use the depthwise separable convolution instead of common convolution, which has greatly reduced the redundancy, but our method can further compress it. As demonstrated in Table 3, with 43% FLOPs reduction, the pruning of FVRCP causes zero performance degradation, exceeding the 0.75× uniform baseline [23] by 2.2%. Under the same FLOPs constraints, FVRCP significantly outperforms the AutoML method NetAdapt [48] by 0.7% but uses much less memory and computational resources.

4.3 Effect of Channel Importance Metric

To demonstrate the effectiveness of the proposed channel importance metric, which avoids the influence of redundant parameter scales, we perform the pruning experiments with it as well as some parameter magnitude-based ones. For the fair comparison, we conduct pruning at single-shot without any fine-tuning. The compared metrics include commonly used absolute value of BN scales [12, 14], ℓ_1, ℓ_2-norm of channel weights [16, 17] and their variants $(\ell_1 + \ell_2)/2$-norm and $\ell_{2,1}$-norm $\|K_{:,i}\|_{2,1} = \sum_{j=1}^{N} \|K_{j,i}\|_F$ (group Lasso regularizer [17]). The statistics



(a) (b) (c)

Fig. 4. (a) Comparison of uniformly scaled ResNet-56 and FVRCP-pruned ones on CIFAR-10. The accuracy and FLOPs of pruning baseline are 93.65% and 125.5M, respectively. (b) ImageNet top-1 accuracies of FVRCP-pruned ResNet-18 with different numbers of pruning epochs. (c) FVRCP pruning results of ResNet-32 with five different baseline accuracies (*shown in the legend*) on CIFAR-10. Each trial is repeated 5 times.

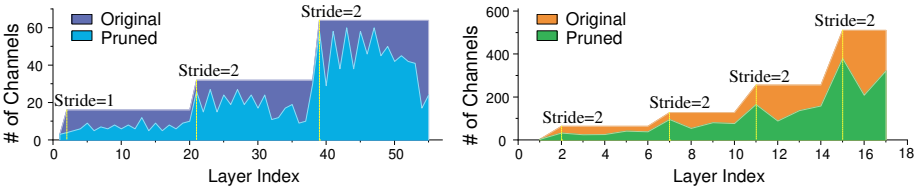


Fig. 5. Pruned architectures of ResNet-56 (*left*) and ResNet-18 (*Right*) with 50% FLOPs reduction.

of input feature maps for SFVR have been calculated on the training set during model training.

The left graph of Fig. 3 shows the comparison results. The pruned accuracies of pre-activation ResNet-20 of SFVR are significantly better than that of parameter magnitude-based ones. It is not surprising since the magnitude of parameters contains redundant information (recall that ResNets employ BN and ReLU activation), and the metrics that are directly based on it cannot assess channel importance accurately. The curve of SFVR which drops slowly at first and then rapidly also illustrates that SFVR can better characterize channel importance (prunes unimportant channels first). The right graph of Fig. 3 compares the single-shot pruning with few fine-tuning epochs with a fixed learning rate of 0.001. As shown, the networks pruned by our metric recover from the pruning more quickly. Even with few fine-tuning epochs, our pruned network can achieve high performance. For more comparisons, please refer to the supplementary.

4.4 Comparison with Uniform Channel Reduction

Uniform channel reduction (uniformly reducing filters in each layer) is commonly used in practice to reduce model size. However, such strategy is based on empirical analysis, failing to achieve the optimal compression ratios. Fig. 4(a) compares the accuracy of uniformly scaled ResNet-56 with FVRCP automatically pruned ones. To ensure the convergence, we train each uniformly scaled ResNet-56 for

600 epochs, much longer than the usually adopted benchmark (160 [1]). The results show that the pruned ResNet-56 model outperforms the uniformly scaled one regardless of FLOPs constraints, resulting in a more efficient architecture.

Neural Architecture Search. In network design, people are curious about what is the best channel allocation policies. Lots of human experts are dedicated to manually designing the channel size of each layer. A recent study [51] argues that the global channel pruning can be viewed as an architecture search method and automatically finds good layer-wise channel numbers. We visualize the pruned architecture by FVRCP, trying to find some design heuristic. Fig. 5 illustrates the pruned architectures of ResNet-56 and ResNet-18 with 50% FLOPs reduction. Interestingly, we find that the channels in the downsampling layers are more retained. It is natural since when there is downsampling, the resolution of feature maps decreases, and thus there should be more channels to prevent the information loss. The same phenomenon has also been observed in [31] in pruned MobileNets, but with ResNets we see more interesting things. We note that the channels in the second layer of the residual block are also more kept. We argue that this mainly results from the shortcut connections by which the input feature maps in the first layer of residual block can be losslessly transferred into later layers while the input feature maps in the second layer cannot (and therefore becomes more important). Moreover, we observe that the higher layers in the last two stages of ResNet-56 are pruned more aggressively. We suspect it is because CIFAR-10 classification is a simple task (only 10 classes) and ResNet-56 is such a deep network that higher layers are underutilized. In contrast to this, for the shallow network ResNet-18 on large-scale ImageNet dataset, we do not observe a similar phenomenon. These facts inspire us to allocate channel numbers in a better way, on which we would do further study in the future.

4.5 Sensitivity Analysis

Number of Pruning Epochs. We change the number of pruning epochs from 10 to 40 on ImageNet on ResNet-18 to explore its influence. As shown in Fig. 4(b), with small FLOPs pruning ratio (25%), increasing pruning epochs does not improve pruned accuracy while with the large one (50%), it does. We attribute this to the smoothness of pruning process. For the large pruning ratio, too few pruning epochs leads to the sharp pruning at the end of each epoch, which may result in irreversible damage to the network. When the pruning epochs increases, pruning process becomes smooth, and the pruned accuracy is then improved.

Pre-trained Model Performance. Fig. 4(c) illustrates FVRCP pruning results of ResNet-32 with five different baseline accuracies. The higher the baseline is, the better the pruned ones are. It is not surprising since FVRCP preserves the principal features in the network and the network that has higher baselines extracts more general and representative features. Also, we note that FVRCP maintains the model performance over a wide range of pruning ratios regardless of original accuracies. It implies that there are indeed a lot of redundancies in commonly trained networks and with FVRCP we can discover and remove them.

4.6 Visualization of Feature Maps

Fig. 6 visualizes the randomly selected five output feature maps in the original and pruned *Block1-Conv2* layer of ResNet-18. The average Pearson correlation coefficient of each layer after pruning is also reported. Consistent with theoretical analysis, the output feature maps before and after pruning by our metric are highly correlated, outperforming Li *et al.* [13] and Liu *et al.* [12] significantly. Li *et al.* employ the ℓ_1-norm of filters to guide the pruning. However, the norms of filters are normalized by the following BN transform. Their values are less helpful in recognizing redundant channels. Liu *et al.* leverage the scaling factors of BN layer to conduct pruning, but the inadequate magnitude of scaling factors can be completely compensated by the following convolution. Useful features will be falsely discarded in these two methods, which severely impair model performance. In contrast, our metric preserves essentially informative channels. It discovers the compact structure embedded in the original convolutions, which improves the representation of feature maps as well as the efficiency of networks.

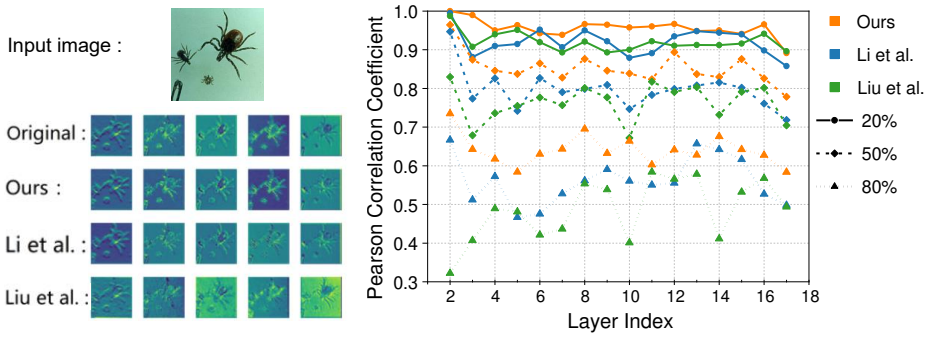


Fig. 6. Visualization of randomly selected five output feature maps in *Block1-Conv2* layer of ResNet-18 as well as those after pruning 50% channels of that layer (*left*). The average Pearson coefficient of each layer between original output feature maps and those after pruning 20%, 50%, and 80% channels of that layer is also reported (*right*).

5 Conclusions

In this paper, we presented a novel channel importance metric based on the Pearson correlation coefficient. The new metric identifies essentially informative channels. Compared with conventional parameter magnitude-based ones, it avoids the influence of redundant parameter scales resulted from the properties of batch normalization and activation layers. Further, we established a channel pruning framework named FVRCP. FVRCP prunes channels globally with little human intervention. It automatically finds important layers in the network. On several benchmarks, FVRCP achieves state-of-the-art results.

References

1. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2016) 770–778
2. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. (2012) 1097–1105
3. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems. (2015) 91–99
4. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2015) 3431–3440
5. Zhang, X., Zou, J., He, K., Sun, J.: Accelerating very deep convolutional networks for classification and detection. IEEE transactions on pattern analysis and machine intelligence **38** (2015) 1943–1955
6. Tai, C., Xiao, T., Zhang, Y., Wang, X., et al.: Convolutional neural networks with low-rank regularization. arXiv preprint arXiv:1511.06067 (2015)
7. Zhou, A., Yao, A., Guo, Y., Xu, L., Chen, Y.: Incremental network quantization: Towards lossless cnns with low-precision weights. arXiv preprint arXiv:1702.03044 (2017)
8. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. In: NIPS Deep Learning and Representation Learning Workshop. (2015)
9. Han, S., Mao, H., Dally, W.J.: Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. arXiv preprint arXiv:1510.00149 (2015)
10. He, Y., Kang, G., Dong, X., Fu, Y., Yang, Y.: Soft filter pruning for accelerating deep convolutional neural networks. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18, International Joint Conferences on Artificial Intelligence Organization (2018) 2234–2240
11. He, Y., Zhang, X., Sun, J.: Channel pruning for accelerating very deep neural networks. In: Proceedings of the IEEE International Conference on Computer Vision. (2017) 1389–1397
12. Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., Zhang, C.: Learning efficient convolutional networks through network slimming. In: Proceedings of the IEEE International Conference on Computer Vision. (2017) 2736–2744
13. Li, H., Kadav, A., Durdanovic, I., Samet, H., Graf, H.P.: Pruning filters for efficient convnets. arXiv preprint arXiv:1608.08710 (2016)
14. Ye, J., Lu, X., Lin, Z., Wang, J.Z.: Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers. In: International Conference on Learning Representations. (2018)
15. Zhuang, Z., Tan, M., Zhuang, B., Liu, J., Guo, Y., Wu, Q., Huang, J., Zhu, J.: Discrimination-aware channel pruning for deep neural networks. In: Advances in Neural Information Processing Systems. (2018) 875–886
16. Li, Y., Lin, S., Zhang, B., Liu, J., Doermann, D., Wu, Y., Huang, F., Ji, R.: Exploiting kernel sparsity and entropy for interpretable cnn compression. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2019) 2800–2809

17. Wen, W., Wu, C., Wang, Y., Chen, Y., Li, H.: Learning structured sparsity in deep neural networks. In: Advances in neural information processing systems. (2016) 2074–2082
18. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE international conference on computer vision. (2015) 1026–1034
19. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Bach, F., Blei, D., eds.: Proceedings of the 32nd International Conference on Machine Learning. Volume 37 of Proceedings of Machine Learning Research., Lille, France, PMLR (2015) 448–456
20. Dinh, L., Pascanu, R., Bengio, S., Bengio, Y.: Sharp minima can generalize for deep nets. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR. org (2017) 1019–1028
21. He, Y., Liu, P., Wang, Z., Hu, Z., Yang, Y.: Filter pruning via geometric median for deep convolutional neural networks acceleration. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2019) 4340–4349
22. Luo, J.H., Wu, J., Lin, W.: Thinet: A filter level pruning method for deep neural network compression. In: Proceedings of the IEEE international conference on computer vision. (2017) 5058–5066
23. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
24. Guo, Y., Yao, A., Chen, Y.: Dynamic network surgery for efficient dnns. In: Advances In Neural Information Processing Systems. (2016) 1379–1387
25. Han, S., Pool, J., Tran, J., Dally, W.: Learning both weights and connections for efficient neural network. In: Advances in neural information processing systems. (2015) 1135–1143
26. Tung, F., Mori, G.: Clip-q: Deep network compression learning by in-parallel pruning-quantization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2018) 7873–7882
27. Zhang, T., Ye, S., Zhang, K., Tang, J., Wen, W., Fardad, M., Wang, Y.: A systematic dnn weight pruning framework using alternating direction method of multipliers. In: Proceedings of the European Conference on Computer Vision (ECCV). (2018) 184–199
28. Dong, X., Chen, S., Pan, S.: Learning to prune deep neural networks via layer-wise optimal brain surgeon. In: Advances in Neural Information Processing Systems. (2017) 4857–4867
29. LeCun, Y., Denker, J.S., Solla, S.A.: Optimal brain damage. In: Advances in neural information processing systems. (1990) 598–605
30. Yu, R., Li, A., Chen, C.F., Lai, J.H., Morariu, V.I., Han, X., Gao, M., Lin, C.Y., Davis, L.S.: Nisp: Pruning networks using neuron importance score propagation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2018) 9194–9203
31. Liu, Z., Mu, H., Zhang, X., Guo, Z., Yang, X., Cheng, K.T., Sun, J.: Metapruning: Meta learning for automatic neural network channel pruning. In: Proceedings of the IEEE International Conference on Computer Vision. (2019) 3296–3305
32. Ding, X., Ding, G., Guo, Y., Han, J.: Centripetal sgd for pruning very deep convolutional networks with complicated structure. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2019) 4943–4953

33. Kim, J., Park, S., Kwak, N.: Paraphrasing complex network: Network compression via factor transfer. In: Advances in Neural Information Processing Systems. (2018) 2760–2769
34. Wang, K., Liu, Z., Lin, Y., Lin, J., Han, S.: Haq: Hardware-aware automated quantization with mixed precision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2019) 8612–8620
35. Zhu, C., Han, S., Mao, H., Dally, W.J.: Trained ternary quantization. arXiv preprint arXiv:1612.01064 (2016)
36. Son, S., Nah, S., Mu Lee, K.: Clustering convolutional kernels to compress deep neural networks. In: Proceedings of the European Conference on Computer Vision (ECCV). (2018) 216–232
37. Chellapilla, K., Puri, S., Simard, P.: High performance convolutional neural networks for document processing. In: International Workshop on Frontiers in Handwriting Recognition. (2006)
38. Chetlur, S., Woolley, C., Vandermersch, P., Cohen, J., Tran, J., Catanzaro, B., Shelhamer, E.: cudnn: Efficient primitives for deep learning. arXiv preprint arXiv:1410.0759 (2014)
39. Zhu, M., Gupta, S.: To prune, or not to prune: exploring the efficacy of pruning for model compression. arXiv preprint arXiv:1710.01878 (2017)
40. Lin, S., Ji, R., Yan, C., Zhang, B., Cao, L., Ye, Q., Huang, F., Doermann, D.: Towards optimal structured cnn pruning via generative adversarial learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2019) 2790–2799
41. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images. Technical report, Citeseer (2009)
42. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision **115** (2015) 211–252
43. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., et al.: Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467 (2016)
44. Lin, M., Ji, R., Wang, Y., Zhang, Y., Zhang, B., Tian, Y., Shao, L.: Hrank: Filter pruning using high-rank feature map. In: CVPR 2020: Computer Vision and Pattern Recognition. (2020) 1529–1538
45. Lin, S., Ji, R., Li, Y., Wu, Y., Huang, F., Zhang, B.: Accelerating convolutional networks via global & dynamic filter pruning. In: IJCAI. (2018) 2425–2432
46. Singh, P., Verma, V.K., Rai, P., Namboodiri, V.: Leveraging filter correlations for deep model compression. In: The IEEE Winter Conference on Applications of Computer Vision. (2020) 835–844
47. Li, Y., Gu, S., Mayer, C., Gool, L.V., Timofte, R.: Group sparsity: The hinge between filter pruning and decomposition for network compression. In: CVPR 2020: Computer Vision and Pattern Recognition. (2020) 8018–8027
48. Yang, T.J., Howard, A., Chen, B., Zhang, X., Go, A., Sandler, M., Sze, V., Adam, H.: Netadapt: Platform-aware neural network adaptation for mobile applications. In: Proceedings of the European Conference on Computer Vision (ECCV). (2018) 285–300
49. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: European conference on computer vision, Springer (2016) 630–645
50. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2017) 4700–4708

51. Liu, Z., Sun, M., Zhou, T., Huang, G., Darrell, T.: Rethinking the value of network pruning. arXiv preprint arXiv:1810.05270 (2018)