



Towards efficient filter pruning via topology

Xiaozhou Xu¹ · Jun Chen¹ · Hongye Su¹ · Lei Xie¹

Received: 25 November 2021 / Accepted: 23 February 2022 / Published online: 19 March 2022
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2022

Abstract

With the development of deep neural networks, compressing and accelerating deep neural networks without performance deterioration has become a research hotspot. Among all kinds of network compression methods, network pruning is one of the most effective and popular methods. Inspired by several property-based pruning methods and geometric topology, we focus the research of the pruning method on the extraction of feature map information. We predefine a metric, called TopologyHole, used to describe the feature map and associate it with the importance of the corresponding filter. In the exploration experiments, we find out that the average TopologyHole of the feature map for the same filter is relatively stable, regardless of the number of image batches the CNNs receive. This phenomenon proves TopologyHole is a data-independent metric and valid as a criterion for filter pruning. Through a large number of experiments, we have demonstrated that priorly pruning the filters with high-TopologyHole feature maps achieves competitive performance compared to the state-of-the-art. Notably, on ImageNet, TopologyHole reduces 45.0% FLOPs by removing 40.9% parameters on ResNet-50 with 75.71%, only a loss of 0.44% in top-1 accuracy.

Keywords Model compression · Filter pruning · Neural networks · Image classification

1 Introduction

In recent years, deep learning technique has been used for data analysis in the fields such as computer vision [1, 2] and natural language processing [3, 4], achieving unprecedented success. Meanwhile, the advantages that Convolutional Neural Networks (CNNs) show in image classification [5–7], detection [8, 9] and segmentation [10, 11] attract more research interests. To improve the fitting ability of the models, the structure of the neural networks becomes deeper, and the number of parameters is increasing. On the one hand, the accuracy of the neural networks for the image recognition task is improved. On the other hand, there is a growing demand for computing and storage resources [12, 13]. The overhead of advanced Graphical Processing Unit (GPU) equipment for training and reasoning limits the usage of complex models, let alone resource-constrained edge devices, such as portable mobile devices and wearable

devices. Previous studies [14–16] have revealed that there is much redundancy in deep networks. Although these redundant parameters ensure the uniformity of model structures, such excessive parameters increase the space and time complexity of the networks, which brings more negative impact on application than positive impact on accuracy. Therefore, various network compression methods have been developed, compromising the tradeoffs between redundant parameters, running time of the models, and accuracy.

Generally, the network compression methods include parameter quantization [15, 17, 18], knowledge distillation [19–21], low-rank approximation [22–24], compact model design [25, 26], network pruning [12, 27] and etc. Quantization maps the network weights to a smaller range of values and storage bits. Knowledge distillation utilizes the knowledge of the teacher network and transfers it to a compact distillation model. The low-rank approximation uses matrix or tensor decomposition techniques to decompose the original convolution filters. Moreover, the compact model design aims to develop a specially structured convolution kernel or compact convolution computing unit to reduce the computational complexity of the model.

Network pruning is well employed on mainstream hardware. It aims at cutting off the redundant weights or filters of

✉ Lei Xie
leix@iipc.zju.edu.cn

¹ State Key Laboratory of Industrial Control Technology, College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China

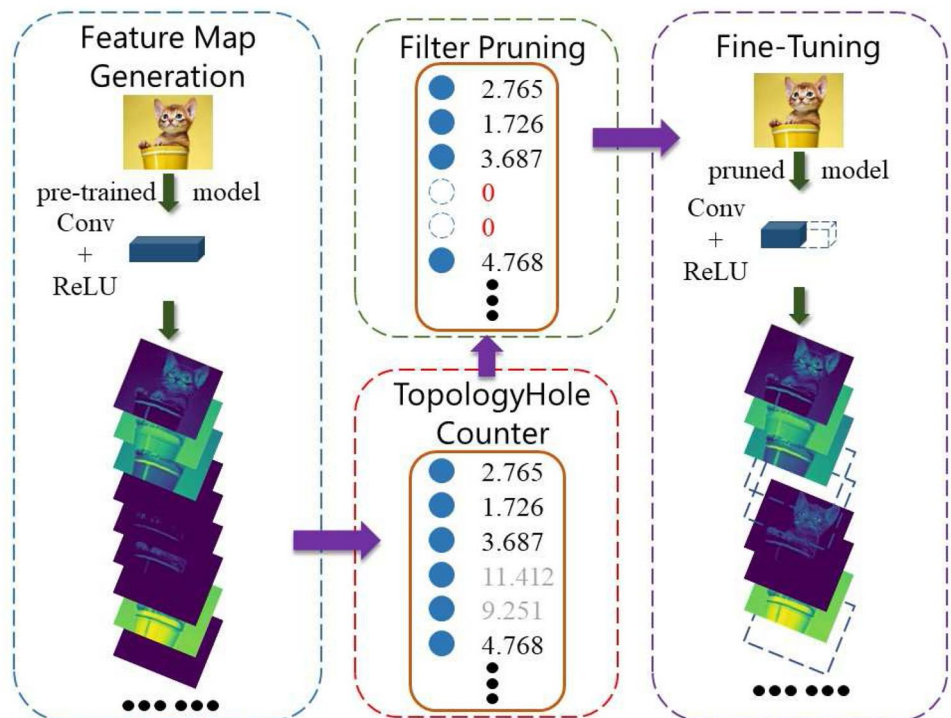
CNNs, along with their associated computations, and generating more compact subnetworks [15, 27, 28]. In contrast to weight-level pruning, filter-level pruning results in models with structured sparsity and only needs the Basic Linear Algebra Subprograms (BLAS) library to improve performance, which is more flexible in practical implementation. In this paper, our research focuses on structured filter pruning. Empirically, we categorize the existing structured filter pruning into three groups: property-based pruning, imposed scaling factor-based pruning, and propagation-based pruning. Compared with the other two groups of methods, the calculation of property-based pruning is more straightforward with no need for additional constraints and factors, so the mainstream filter pruning methods [27, 29–31] mainly belong to this group.

However, most property-based pruning methods are denoted to establishing the connection between the weights of the convolutional layers and the importance of filters and prune filters through the smaller-norm-less-important criterion. This criterion requires that the deviation of filter norms should be significant and the norms of filters to be pruned are expected to be absolutely small, i.e., close to zero. These two prerequisites greatly limit the effectiveness of norm-based methods. In this case, we develop a feature map-based pruning method. The feature map is generated after the convolutional layer and ReLU layer. It is a one-to-one correspondence with the filter of the convolutional layer. In our opinion, the information of the feature maps better represents the importance of the information contained in

the corresponding filters than other intrinsic properties of the pre-trained model, such as the weights of the convolutional layers. By analyzing APoZ [30], the most popular pruning method based on feature maps, we find that simply calculating the percentage of zero activations of the feature map cannot sufficiently represent the importance of the filter corresponding to the feature map. We attribute this problem to its data-driven scheme and heavily relying on the input distribution, which is unstable on effect and not conducive to deployment. Instead, a data-independent feature map-based filter pruning scheme is more convincing.

To explore the inherent characteristics of feature maps, we introduce the concept of geometric topology. Topology is the study of properties of geometric figures or spaces that remain the same after continuous changes in shape. Genus is a scalar used in topology to distinguish between topological spaces. In this paper, we propose the TopologyHole method, which defines a new metric referring to the genus to evaluate the information of the feature map, as shown in Fig. 1. We find that according to the definition of TopologyHole, the mean value of TopologyHole of the feature map generated by the same filter is relatively stable no matter how the data for CNN is distributed, as shown in Fig. 2. This phenomenon implies that TopologyHole represents implicit information of the feature maps, which can be used as the criterion for pruning by filters. Based on this, the filters generating feature maps with high TopologyHole can be removed because we confirm that feature maps with high TopologyHole have greater sparsity and more redundant information. We have

Fig. 1 Framework of TopologyHole. The left column shows the normal pre-trained model inference process. Each group of convolutional layer and ReLU activation layer will generate a set of feature maps. In the middle column, we calculate each set of feature maps according to the definition of TopologyHole, sort the feature maps generated in the same layer, and remove the filters with high TopologyHole. In the right column, we fine-tune the pruned model to recover the influence of pruning on the accuracy



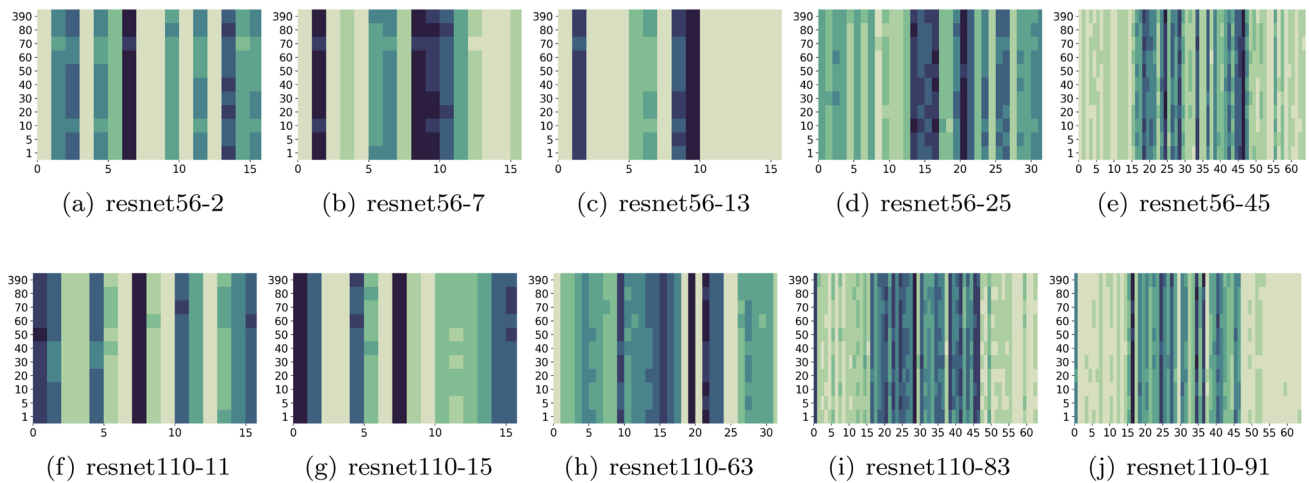


Fig. 2 Average TopologyHole of feature maps from different convolutional layers and architectures on CIFAR-10. Specifically, the subtitles of the subgraph describe the indexes of feature maps to which the extracted TopologyHole belongs, mapped by ReLU after the convolutional layer of the models. The abscissa of the subgraph represents

the indices of the feature maps of the current layer, corresponding to the indices of filters of the convolutional layer simultaneously. The ordinate is the batches of training images (each batch size is set to 128). Different colors represent different TopologyHole numerical sizes

proved that high-TopologyHole feature maps have a low contribution to the model prediction accuracy through many experiments.

We conducted many pruning experiments with different models, including VGG-16 [13], GoogLeNet [6] and ResNet [5], on two benchmark datasets, CIFAR-10 [32] and ImageNet [33]. The results show that our TopologyHole leads the state-of-the-art performance better than the existing filter pruning methods. We also carry out experiments to show that the filters generating feature maps with high TopologyHole can be removed more reasonably than those with low TopologyHole.

To summarize, our main contributions are:

- We define a novel metric, TopologyHole, to describe the amount of information contained in the feature map. As a data-independent metric, TopologyHole is friendly to deployment because it does not depend on the input data distribution.
- Compared with the most popular feature map-based pruning method, APoZ, our TopologyHole takes the position relation of zero activations in the feature map into account, which is more convincing than APoZ.
- Through a large number of experiments, we confirm that feature maps with high Topology are less critical to preserving accuracy. Thus, the filters generating these feature maps can be removed first.
- Extensive experiments demonstrate the efficacy of our TopologyHole in making tradeoffs between model complexity and accuracy. Our proposed method achieves better performances with similar compression rates in

network compression over the various state-of-the-art [29, 31, 34–38].

The rest of the paper is organized as follows. Section 2 summarizes related works for network compression and classifies them into three groups. Section 3 describes the details of the proposed pruning method. The experimental settings and results are illustrated and discussed in Sect. 4. In addition, Sect. 5 gives a conclusion.

2 Related work

Filter pruning is a structural pruning method that discards the whole filters in current convolutional layers. The core of the filter pruning method lies on how to quantify the importance of each filter in the network. Based on the features where filter selection strategy extracts evaluation metrics, we categorize the existing filter pruning approaches into the following three groups:

- (1) Property-based pruning: The filters are pruned by a specific property of the pre-trained model. Li et al. [27] calculated the L_1 -norm of each filter in the convolutional layers and believed that filters with small norms could be pruned first because they were less important. He et al. [29] calculated the geometric median of the filters within the same layer and pruned filters closest to each other. Lin et al. [38] encoded the second-order information of pre-trained weights, which enabled the representation capacity of pruned networks to be

recovered with a simple finetuning procedure. Hu et al. [30] pruned filters corresponding to the feature maps mapped by ReLU with a high percentage of zero activations. Lin et al. [31] evaluated the rank of feature maps and pruned filters generating low-rank feature maps.

- (2) Imposed scaling factor-based pruning: Huang et al. first introduced imposed scale factor to scale the outputs of specific model structures [34]. Lin et al. [37] introduced the idea of generative adversarial learning and removed the basic structures, including channels, branches, or blocks, by the sparsification of the soft mask. Tian et al. [39] introduced a trainable collaborative layer to prune and learn neural networks in one go jointly.
- (3) Propagation-based pruning: Luo et al. [35] utilized the input of the next layer of the convolution layer to guide pruning the output channel of the current layer. Yu et al. [36] used the proposed propagation algorithm to push the importance scores of final responses back layer by layer and pruned the filters with the least importance. Molchanov et al. [40] defined the filter importance as the change of loss caused by removing a specific filter from the network and used a Taylor expansion to approximate it. Lian et al. [41] introduced an evolutionary algorithm into the process of searching for the most suitable number of pruned filters for every layer.

Discussion The imposed scaling factor-based pruning methods inevitably introduce additional computation, and the propagation-based pruning methods are always data-dependent, whose utilizing of training data is computationally intensive. Therefore, we focus on property-based pruning methods. Among the property-based methods, traditional smaller-norm-less-important criterion [27] is not convincing that two prerequisites limit its usage (mentioned in Sect. 1). Considering that the feature maps correspond to the convolutional layer filters, some researchers have begun to explore the possibility of pruning according to the feature maps. To the best of our knowledge, APoZ [30] was the most popular method to prune the network through feature maps. We highlight our advantages compared to this approach as follows: (1) based upon extensive statistical validations, we empirically demonstrate that the mean value of TopologyHole of feature map generated by the same filter is relatively stable. Therefore, our TopologyHole is a data-independent metric that is friendly to deployment because it does not depend on the input data distribution, unlike APoZ; (2) our TopologyHole considers the percentage of zero activations of the feature map like APoZ and the position relation of zero activations; (3) the experiments demonstrate our TopologyHole is more effective than APoZ with more FLOPs reduction on CIFAR-10 shown in Table 4.

The following section of the paper introduces the mathematical definition of TopologyHole and its efficacy in filter pruning for modern advanced CNN models on two benchmark datasets, CIFAR-10 and ImageNet.

3 Methodology

3.1 Preliminaries

We assume a pre-trained CNN network has L convolutional layers. Note that a convolutional layer here contains pooling, batch normalization and ReLU activation. We use the *thop* library to calculate the FLOPs and the amount of parameters of the network. The deep CNN network can be parameterized by $W_{L^i} = \{w_1^i, w_2^i, \dots, w_{N_{i+1}}^i\} \in \mathbb{R}^{N_{i+1} \times N_i \times K_i \times K_i}$, where i denotes the i -th convolutional layer, N_{i+1} is the number of filters as well as N_i is the number of input channel in L^i , and K_i represents the kernel size. When the image data is fed into the model, each filter $w_j^i \in \mathbb{R}^{N_i \times K_i \times K_i}$ outputs a feature map $o_j^i \in \mathbb{R}^{h_i \times w_i}$, $1 \leq j \leq N_{i+1}$, where h_i and w_i are the height and width of the feature map, respectively.

In the pruning phase, we first determine the number of feature maps to be removed by the number of output channels N_{i+1} of the current layer and pruning rate P_i . Then, rank the feature maps according to the importance of the defined evaluation strategy, and remove the corresponding least important $N_{i+1} \times P_i$ filters. At the same time, the input tensor of the L^{i+1} layer is also smaller, further reducing the FLOPs and the number of parameters.

3.2 TopologyHole

The difference between network pruning methods for manually determining pruning rate lies in the criteria for judging unimportant filters of each layer. Compared with other pruning methods, property-based pruning is more focused on a specific property of the pre-trained model, e.g. percentage of zero activations or rank of feature maps or L_p -norm for filters, and prunes the corresponding filters with less importance.

We define TopologyHole to measure the information richness of feature maps:

$$S_j^i(I) = \frac{1}{\text{batch}} \sum_{\text{num}=1}^{\text{batch}} \text{TopologyHole}(o_j^i)_{\text{num}}, \quad (1)$$

where $\text{TopologyHole}(\cdot)$ is the TopologyHole of a feature map for input data I . S_j^i is the measurement (i.e., the number of TopologyHole) for the j -th filter of the i -th convolutional layer and batch is the number of input images.

We confirm that the filter with a larger TopologyHole contains less information. Specifically, the count of TopologyHole conforms to the following criteria:

- In the feature map matrix of $h_i \times w_i$ size output by the L^i convolutional layer, the feature cell with weight of 0 will be counted as one TopologyHole as shown in Fig. 3-①.
- If many feature cells with a weight of 0 are connected, they will be collectively counted as one TopologyHole, and individual cells are not repeatedly counted, as shown in Fig. 3-②. Note that diagonals touching cells are not considered connected.
- If any of the connected feature cells whose weights are all 0 are located in the outermost circle of the feature map matrix, the number of TopologyHole is not considered for the connected cells as a whole, as shown in Fig. 3-③

Therefore, we conclude that there are two TopologyHoles for the two feature maps in Fig. 3 by counting TopologyHole using the above three principles.

3.3 Tractability

As we mentioned above, we find that according to the definition of TopologyHole, the mean value of TopologyHole of the feature map generated by the same filter is relatively stable no matter how the data given to CNNs is distributed, as shown in Fig. 2. Therefore, we can use a small batch of image data to calculate the average TopologyHole of each feature map.

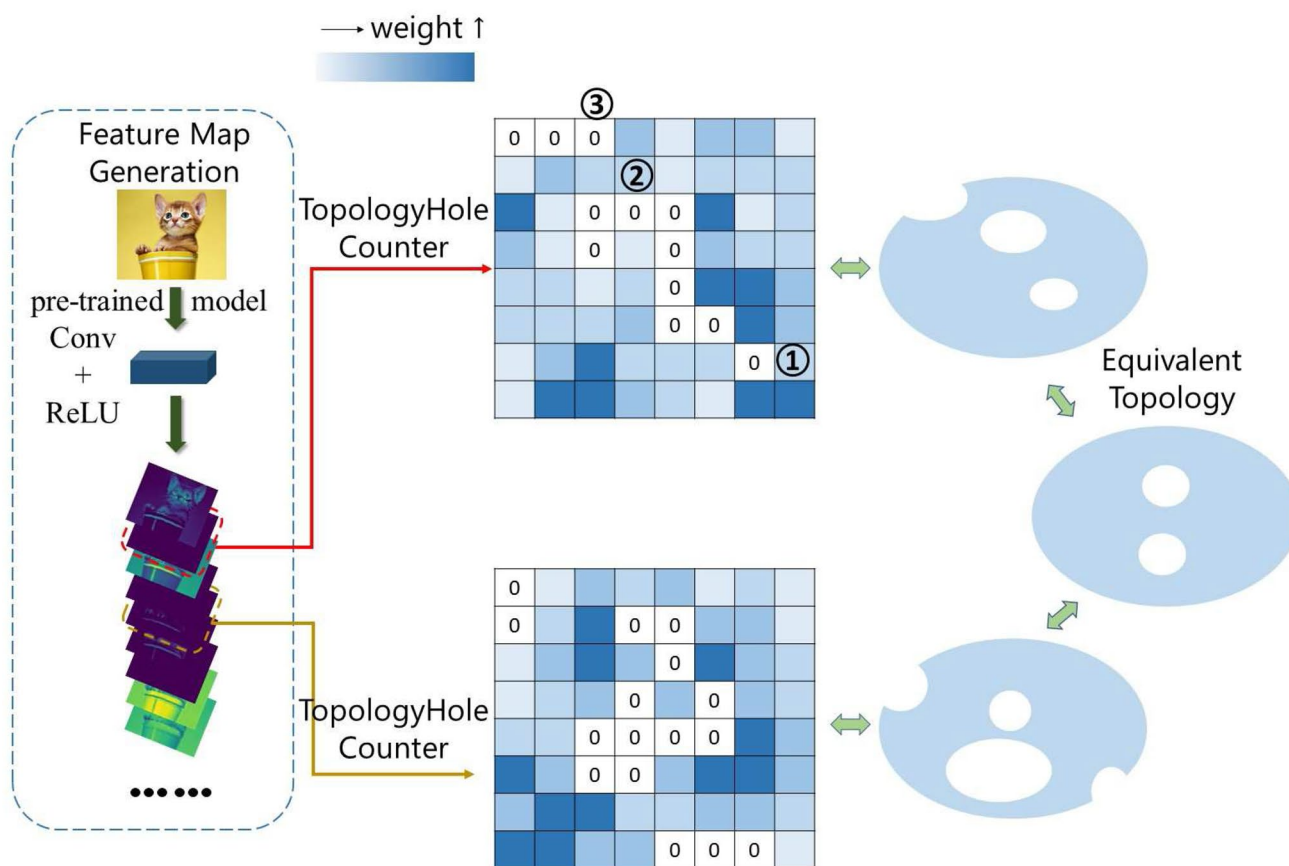


Fig. 3 TopologyHole Counting Standard and Equivalent Topology Structure. The upper and lower feature maps are extracted from the same (Conv+ReLU) layer output by the neural network. They have different weight distributions, but according to our TopologyHole

counting standard, they have the same number of Topologyholes, which means they are equivalent to the right-center topology structure with two TopologyHoles

Algorithm 1 TopologyHole Counter

Require: Feature map o_j^i output by the j -th filter of L^i .
Ensure: $pointer$.

```

1: for  $m = 0; m < h_i; m++$  do
2:   for  $n = 0; n < w_i; n++$  do
3:     if  $o_j^i(m, n) = 0$  then
4:        $pointer = pointer + 1$ .
5:        $o_j^i(m, n) = pointer$ .
6:        $p = m, q = n$ 
7:       while  $o_j^i(x, y) = 0$  and  $x = p \pm 1$  and  $y = q \pm 1$ 
8:         do
9:            $o_j^i(x, y) = pointer$ .
10:           $p = x, q = y$ 
11:        end while
12:      end if
13:    end for
14:  end for
15: end for

```

Algorithm 2 TopologyHole Pruning Algorithm

Require: Pre-trained model F with filter set $W = \{W_{L^i}, 1 \leq i \leq L\}$, training data X , pruning rate P_i .
Ensure: Pruned model F' with filter set W' .

```

1: for  $i = 1; i \leq L; i++$  do
2:   Calculate  $S_j^i$  for each filter by Eq. 1 and Alg. 1.
3:   Remove  $N_{i+1} \times P_i$  filters with highest-TopologyHole.
4: end for
5: Initialize  $F'$  with pre-trained model weights.
6: for  $epoch = 1; epoch \leq epoch_{max}; epoch++$  do
7:   Fine-tune the pruned model  $F'$ .
8: end for
9: Return  $F'$  with fine-tuned filter set  $W'$ .

```

To illustrate that pruning using TopologyHole is very effective, we summarize the TopologyHole counting method and TopologyHole pruning process in Algorithm 1 and Algorithm 2, respectively.

In Algorithm 1, we input the feature map o_j^i of the j -th filter of L^i into the TopologyHole counter. The next phase is to traverse each pixel of o_j^i . If its pixel value is not 0, it skips and detects the next pixel. On the contrary, if its pixel value is 0, the current TopologyHole count will be reassigned to it, and p and q are used to store the current pixel position m and n , respectively. When the pixel value of the four points connected by pixel point (p, q) is 0, the pixel value is repeatedly assigned as the current TopologyHole count value. Simultaneously, extended detection and assignment are carried out to the connected pixels of pixels with pixel value 0 until all connected pixels with the same value of 0 are reassigned with the same

TopologyHole count value, and then the TopologyHole count will be updated, and the next pixel is detected as above.

The details of the TopologyHole pruning procedure are illustratively explained in Algorithm 2. Before selecting pruning filters, the training data X and the pruning rate P_i of each layer should be input into the pre-trained full model. Then, the TopologyHole of each filter in the same convolutional layer is calculated as S_j^i by Eq. 1 Algorithm 1. Sort $\{S_j^i, 1 \leq j \leq N_{i+1}\}$ and remove the $N_{i+1} \times P_i$ filters with highest TopologyHole. Thus, a pruning model inheriting the weight of the full pre-trained model is obtained. The precision loss caused by pruning will be recovered by retraining the pruned model with an appropriate number of epochs, and the desired pruned model is finally obtained.

4 Experiments

4.1 Experimental settings

Models and dataset To illustrate that our TopologyHole pruning method compresses and accelerates the model effectively for both small and large datasets, we choose CIFAR-10 [32], and ImageNet [33] as two benchmarks. Meanwhile, we compare the pruning performance of our method with that of state-of-the-art (SOTA) on several mainstream models, including VGG-16, ResNet-56/110, GoogLeNet on CIFAR-10, ResNet-50 on ImageNet.

Implementation details We use Pytorch 1.7.1 to implement our TopologyHole filter pruning method under Inter(R) Core(TM) i7-9700K CPU 3.60 GHz and two NVIDIA RTX 3090 with 24 GB for GPU processing. Our TopologyHole method is a type of one-shot pruning approach, and we adopt the Stochastic Gradient Descent algorithm (SGD) with an initial learning rate of 0.01 and 0.1, respectively, for CIFAR-10 and ImageNet as the optimization function. For CIFAR-10, the pruned model is fine-tuned by 300 epochs (reducing learning rate in the 150-th and 225-th epoch) for ResNet-56/110 and GoogLeNet, and 150 epochs (reducing learning rate in the 50-th and 100-th epoch) for VGG-16. The batch size, momentum, and weight decay for the above four model architectures are set to 256, 0.9, and 0.005, respectively. For ImageNet, the pruned model is fine-tuned by 90 epochs reducing the learning rate in the 30-th, 60-th and 80-th epoch. The batch size and momentum of the ImageNet are the same as those of CIFAR-10, except for weight decay set to 0.0001. For a fair and accurate comparison, we utilize the built-in function *thop.profile* to calculate FLOPs and Params.

Evaluation metrics For a fair comparison with other algorithms, we measure the top-1 accuracy of the pruned

Table 1 Results of ResNet-56 on CIFAR-10

Criterion	Top-1%	FLOPs (PR)	Params (PR)
ResNet-56	93.26	125.49 M(0.0)	0.85 M(0.0)
L_1 (2016) [27]	93.06	90.90 M(27.6)	0.73 M(14.1)
HRank (2020) [31]	93.53	90.90 M(27.6)	0.71 M(16.8)
TopologyHole (ours)	94.12	90.35 M(28.0)	0.66 M(22.3)
NISP (2018) [36]	93.01	81.00 M(35.5)	0.49 M(42.4)
GAL (2019) [37]	92.98	78.30 M(37.6)	0.75 M(11.8)
FilterSketch (2021) [38]	93.19	73.36 M(41.5)	0.50 M(41.2)
PBT (2021) [39]	93.12	N/A(43.09)	N/A(47.2)
TopologyHole (ours)	93.76	65.94 M(47.4)	0.48 M(42.8)

model on CIFAR-10 and top-1 and top-5 accuracy on ImageNet. On the other hand, we also calculate the Floating point operations (FLOPs), the number of parameters (Params), and the pruning rate (PR) relative to the full model, to evaluate the pruning effectiveness of the different pruning criteria. FLOPs represents the computational speed of the model, i.e., the time complexity of the network, and Params represents the size of the model, i.e., the space complexity of the network.

4.2 Comparison on CIFAR-10

To prove that our proposed TopologyHole filter pruning method is suitable for pruning different model architectures and shows excellent performance, compression experiments are carried out on several models, including ResNet-56/110, VGG-16, and GoogLeNet. We report the top-1 accuracy of the model pruned by each method. Different pruning methods adopt different FLOPs baselines and Params baselines. If the baselines in the original papers are different from those in this paper, their specific values in the tables will be replaced by N/A, while the drop percentages (i.e. PR, pruning rate) are retained. If the papers to which the methods belong did not have the same evaluation metrics as ours, the vacant metrics are also replaced by N/A. The results of our proposed method are in bold. All results are presented in Tables 1, 2, 3 and 4 on the CIFAR-10 dataset. M/B means million/billion, respectively, and the entries are sorted according to FLOPs (PR)

ResNet-56 Results for ResNet-56 are presented in Table 1. Our TopologyHole removes around 47.4% FLOPs and 42.8% parameters while obtaining the top-1 accuracy at 93.76%. Compared to 93.26% by the original full model, the accuracy is improved by 0.50%. Compared with other property-based pruning methods, HRank (2020) [31] and FilterSketch (2021) [38], TopologyHole shows an overwhelming superiority.

ResNet-110 Table 2 shows that TopologyHole outperforms the SOTAs in both accuracy and time complexity

Table 2 Results of ResNet-110 on CIFAR-10

Criterion	Top-1%	FLOPs (PR)	Params (PR)
ResNet-110	93.50	252.89M(0.0)	1.72M(0.0)
L_1 (2016) [27]	93.30	155.00M(38.7)	1.16M(32.6)
GAL (2019) [37]	92.55	130.20M(48.5)	0.95M(44.8)
PBT (2021) [39]	93.84	N/A(49.6)	N/A(49.4)
SST (2021) [41]	93.62	106.04M(50.6)	N/A
FPGM (2019) [29]	93.74	120.63M(52.3)	N/A
HRank (2020) [31]	93.36	105.70M(58.2)	0.70M(59.2)
TopologyHole (ours)	94.12	101.97M(59.6)	0.72M(58.1)
FilterSketch (2021) [38]	93.44	92.84M(63.3)	0.69M(59.9)
TopologyHole (ours)	93.59	71.69M(71.6)	0.54M(68.3)

Table 3 Results of VGG-16 on CIFAR-10

Criterion	Top-1%	FLOPs (PR)	Params (PR)
VGG-16	93.96	313.73M(0.0)	14.98M(0.0)
FPGM (2019) [29]	93.54	N/A(34.2)	N/A
L_1 (2016) [27]	93.40	206.00M(34.3)	5.40M(64.0)
GAL (2019) [37]	92.03	189.49M(39.6)	3.36M(77.6)
SSS (2018) [34]	93.02	183.13M(41.6)	3.93M(73.8)
HRank (2020) [31]	93.43	145.61M(53.5)	2.51M(82.9)
TopologyHole (ours)	93.93	131.17M(58.1)	2.76M(81.6)

Table 4 Results of GoogLeNet on CIFAR-10

Criterion	Top-1%	FLOPs (PR)	Params (PR)
GoogLeNet	95.05	1.52B(0.0)	6.15M(0.0)
GAL (2019) [37]	93.93	0.94B(38.2)	3.12M(49.3)
APoZ (2016) [30]	92.11	0.76B(50.0)	2.85M(53.7)
HRank (2020) [31]	94.07	0.45B(70.4)	1.86M(69.8)
TopologyHole (ours)	94.89	0.40B(73.7)	2.10M(65.9)

reduction (i.e., FLOPs reduction). Specifically, with 71.6% FLOPs reduction and 68.3% parameters reduction which are much more than those of FilterSketch and HRank, TopologyHole achieves 93.59% top-1 accuracy, 0.15% better than FilterSketch and 0.23% better than HRank. Compared with other SOTAs, like FPGM (2019) [29], TopologyHole with more FLOPs reduction (59.6% of TopologyHole vs. 52.3% of FPGM) obtains 94.12% top-1 accuracy, 0.38% better than that of FPGM.

VGG-16 Table 3 displays the pruning results of VGG-16. TopologyHole reduces the FLOPs of VGG-16 by 58.1% and the parameters by 81.6% while obtaining the top-1 accuracy at 93.93%, only losing 0.03% accuracy relative to the original full model. TopologyHole significantly outperforms other SOTAs.

GoogLeNet For GoogLeNet, as shown in Table 4. TopologyHole removes 73.7% FLOPs and 65.9% parameters with a negligible accuracy drop (94.89% for TopologyHole vs. 95.05% for the baseline). It is significantly better than APoZ (2016) [30] and HRank, which are both pruning based on feature maps like our TopologyHole. Especially for APoZ, TopologyHole achieves better performance among both accuracy and pruning rate (94.89% by TopologyHole vs. 92.11% by APoZ for accuracy, 73.79% by TopologyHole vs. 50.0% by APoZ for FLOPs, 65.9% by TopologyHole vs. 53.7% by APoZ for Params). It proves that although both our TopologyHole and APoZ involve zero activations of feature maps, the TopologyHole counting criterion is completely different from APoZ (see the details in Sect. 3.2). The TopologyHole of feature maps can better serve as a discriminative property for identifying the redundant filters.

4.3 Comparison on ImageNet

We also explore the performance of our proposed TopologyHole filter pruning method on the ImageNet dataset with ResNet-50, a popular CNN. Comparison of pruning

ResNet-50 on ImageNet by our TopologyHole and other effective pruning criteria are shown in Table 5. We adopt top-1 and top-5 accuracy, FLOPs, and parameters reduction as evaluation metrics. We also report the accuracy gap between pruned model and the original full model. For FLOPs and parameters, if the baselines in the methods original papers are different from those in this paper, their specific values in the table will be replaced by N/A, while the drop percentages (i.e., PR, pruning rate) are retained. If the papers to which the methods belong did not have the same evaluation metrics as ours, the vacant metrics are also replaced by N/A. The results of our proposed method are in bold. M/B means million/billion, respectively, and the entries are sorted according to FLOPs (PR). Original performance of full ResNet-50 on ImageNet is 76.15% of top-1 accuracy and 92.87% of top-5 accuracy with 4.11 billion FLOPs and 25.55 million of Params. Compared with other pruning criteria, our TopologyHole performs better in all aspects. Specifically, with 45.0% FLOPs and 40.9% Params reduction, our TopologyHole pruning method achieves 75.71% top-1 accuracy and 92.66% top-5 accuracy while compressing 1.82× of FLOPs and 1.69× of Params.

Table 5 Results of ResNet-50 on ImageNet

ResNet-50							
Criterion	Top-1%	Gap%	Top-5%	Gap%	FLOPs (PR)	Speed up ^a	Params (PR)
Baseline	76.15	–	92.87	–	4.11B(0.0)	1.00×	25.55M(0.0)
SSS-32 (2018) [34]	74.18	–1.97	91.91	–0.96	2.82B(31.0)	1.46×	18.60M(27.0)
He et al. (2017) [28]	72.30	3.85	90.80	–2.07	2.73B(33.6)	2.73×	N/A
ThiNet-70 (2017) [35]	72.04	–4.11	90.67	–2.20	N/A (36.8)	1.58×	16.94M(33.7)
FPGM (2019) [29]	75.59	–0.56	92.63	–0.24	N/A(42.2)	1.73×	N/A
PBT (2021) [39]	74.80	–1.35	N/A	–	2.35B(42.8)	1.74×	13.0M(49.0)
GAL-0.5 (2019) [37]	71.95	–4.20	90.94	–1.93	2.33B(43.0)	1.76×	21.20M(17.0)
SSS-26 (2018) [34]	71.82	–4.33	90.79	–2.08	2.33B(43.0)	1.76×	15.60M(38.8)
HRank-74 (2020) [31]	74.98	–1.17	92.33	–0.54	2.30B(44.0)	1.79×	16.15M(36.8)
TopologyHole (ours)	75.71	– 0.44	92.66	– 0.21	2.26B(45.0)	1.82×	15.09M(40.9)
Taylor (2019) [40]	74.50	–1.65	N/A	–	2.25B(45.3)	1.83×	14.2M(44.5)
FilterSketch-0.6 (2021) [38]	74.68	–1.47	92.17	–0.70	2.23B(45.5)	1.84×	14.53M(43.0)
AutoPruner (2020) [42]	74.76	–1.39	92.15	–0.72	N/A(51.2)	2.05×	N/A
TopologyHole (ours)	74.96	– 1.19	92.37	– 0.50	1.99B(51.6)	2.07×	13.66M(46.5)
GAL-0.5-joint (2019) [37]	71.80	–4.35	90.82	–2.05	1.84B(55.2)	2.23×	19.31M(24.4)
ThiNet-50 (2017) [35]	71.01	–5.14	90.02	–2.85	N/A(55.8)	2.26×	12.38M(51.6)
GAL-1 (2019) [37]	69.88	–6.27	89.75	–3.12	1.58B(61.6)	2.60×	14.67M(42.6)
HRank-71 (2020) [31]	71.98	–4.17	91.01	–1.86	1.55B(62.3)	2.65×	13.77M(46.1)
TopologyHole (ours)	74.16	– 1.99	91.9	– 0.97	1.52B(63.0)	2.70×	11.05M(56.8)
ThiNet-30 (2017) [35]	68.42	–7.73	88.30	–4.57	N/A(71.5)	3.51×	8.66M(66.1)
GAL-1-joint (2019) [37]	69.31	–6.84	89.12	–3.75	1.11B(73.0)	3.70×	10.21M(60.0)
HRank-69 (2020) [31]	69.10	–7.05	89.58	–3.29	0.98B(76.2)	4.19×	8.27M(67.6)
TopologyHole (ours)	72.11	– 4.04	90.81	– 2.06	0.95B(76.9)	4.33×	8.02M(68.6)

^aThe speed up ratio is a theoretical value computed by FLOPs. It is fair to be compared in a same model structure

4.4 Ablation study

We conduct additional ablation studies to analyze further the effectiveness and universality of the TopologyHole filter pruning method.

Filter selection criteria We conduct some experiments on ResNet-56/110 and VGG-16 to demonstrate that it is more appropriate to prune off the filters with highest TopologyHole than lowest TopologyHole. The TopologyHole pruning variants we are comparing include (1) high TopologyHole, i.e., pruning filters with the highest TopologyHole, which is the principle followed by our TopologyHole filter pruning method; (2) low TopologyHole, i.e., pruning filters with lowest TopologyHole. With the same pruning rate of FLOPs and Params, the performance of the model obtained by high TopologyHole pruning is better than that obtained by low-TopologyHole pruning, as shown in Fig. 4. To highlight

the effectiveness of the TopologyHole, we also add seven SOTAs. Among them, APoZ (2016) and HRank (2020) are feature map-based methods like our TopologyHole, while FPGM (2019) and FilterSketch (2021) are other property-based methods. To compare with the other two groups’ structural pruning methods, we also select the latest works to compare (GAL (2019) and PBT (2021) for imposed scaling factor-based pruning method, SST (2021) for propagation-based pruning method).

Varying pruning rates To comprehensively understand the TopologyHole filter pruning method, we test the accuracy of different filter pruning rates for ResNet-56 and the accuracies of the pruned models w.r.t. the variety of filter pruning rates are shown in Fig. 5. With the filter pruning rate increasing, the accuracy of the TopologyHole model first rises above the baseline model and then drops approximately linearly. It should be noted that when we were drawing this

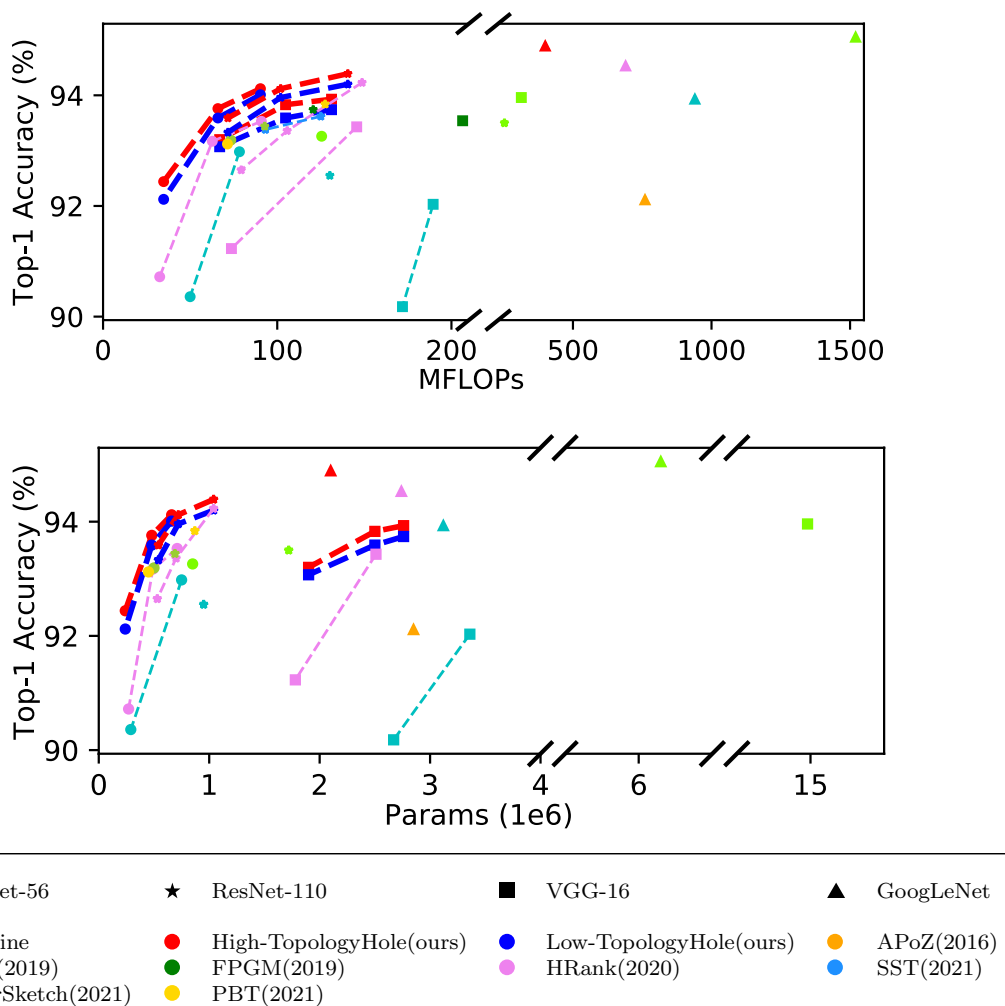


Fig. 4 Comparisons between accuracy and FLOPs (top) and accuracy and remained Parames (bottom) with baseline, four network architectures (ResNet-56, ResNet-110, VGG-16, GoogLeNet) pruned

by seven SOTAs, high TopologyHole and low TopologyHole on the CIFAR-10 dataset. Top-left has the better performance

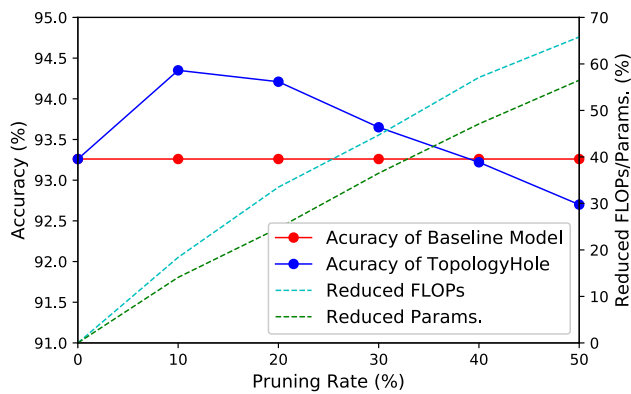


Fig. 5 Accuracies and FLOPs/parameters of the pruned ResNet-56 by TopologyHole w.r.t. the variety of filter pruning rates on CIFAR-10

line chart, we set the filter pruning rate of each layer of the model to be the same, which is obviously not the optimal choice for the pruning rate of each layer, so the accuracies of pruned models are not comparable to that shown in Table 1.

5 Conclusion

Inspired by geometric topology, this paper proposes a novel network pruning method called TopologyHole filter pruning. We predefine a measurement, called TopologyHole, used to describe the feature map and associate it with the importance of the corresponding filter. First of all, we find out that the average TopologyHole of the feature map for the same filter is relatively stable, which means TopologyHole is valid as a metric for filter pruning. Then, through a large number of experiments, we have demonstrated that priorly pruning the filters with high-TopologyHole feature maps achieves competitive performance compared to the state-of-the-art. Specifically, with 45.0% FLOPs and 40.9% parameters reduction, our TopologyHole pruning method achieves 75.71% top-1 accuracy, only 0.44% drop according to the original full model, and 92.66% top-5 accuracy while compressing 1.82× of time complexity and 1.69× of space complexity. Therefore, our proposed TopologyHole filter pruning shows excellent results in reducing the time and space complexity of the network and in reducing the loss of precision of the model after pruning.

References

- 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*, vol. 28, pp. 91–99 (2015)
- He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2961–2969 (2017)
- Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q.V., Salakhutdinov, R.: Transformer-XL: attentive language models beyond a fixed-length context. In: *Association for Computational Linguistics* (2019)
- Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, vol. 1 (Long and Short Papers), pp. 4171–4186 (2019)
- 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*, pp. 770–778 (2016)
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9 (2015)
- Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. *Commun. ACM* **60**(6), 84–90 (2017)
- Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587 (2014)
- Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: Overfeat: integrated recognition, localization and detection using convolutional networks. In: *ICLR* (2014)
- Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440 (2015)
- Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241. Springer, Berlin (2015)
- Han, S., Pool, J., Tran, J., Dally, W.: Learning both weights and connections for efficient neural network. *Adv. Neural Inf. Process. Syst.* **28**, 1135–1143 (2015)
- Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: *ICLR* (2015)
- Denil, M., Shakibi, B., Dinh, L., Ranzato, M., De Freitas, N.: Predicting parameters in deep learning. In: *Advances in Neural Information Processing Systems*, pp. 2148–2156 (2013)
- Han, S., Mao, H., Dally, W.J.: Deep compression: compressing deep neural networks with pruning, trained quantization and Huffman coding. In: *ICLR* (2016)
- Wen, W., Wu, C., Wang, Y., Chen, Y., Li, H.: Learning structured sparsity in deep neural networks. In: *Advances in Neural Information Processing Systems*, pp. 2074–2082 (2016)
- Chen, W., Wilson, J., Tyree, S., Weinberger, K., Chen, Y.: Compressing neural networks with the hashing trick. In: *International Conference on Machine Learning*, PMLR, pp. 2285–2294 (2015)
- Chen, J., Liu, L., Liu, Y., Zeng, X.: A learning framework for n-bit quantized neural networks toward FPGAs. *IEEE Trans. Neural Netw. Learn. Syst.* **32**(3), 1067–1081 (2020)
- Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. In: *NIPS* (2014)
- Z. Li, Y. Sun, G. Tian, L. Xie, Y. Liu, H. Su, Y. He, A compression pipeline for one-stage object detection model. *J. Real Time Image Process.* **18**, 1949–1962 (2021)
- Meng, B., Wang, L., He, Z., Jeon, G., Dou, Q., Yang, X.: Gradient information distillation network for real-time single-image

- super-resolution. *J. Real Time Image Process.* **18**(2), 333–344 (2021)
22. Lebedev, V., Ganin, Y., Rakhuba, M., Oseledets, I., Lempitsky, V.: Speeding-up convolutional neural networks using fine-tuned CP-decomposition. In: *ICLR* (2015)
 23. Denton, E., Zaremba, W., Bruna, J., LeCun, Y., Fergus, R.: Exploiting linear structure within convolutional networks for efficient evaluation. In: *Advances in Neural Information Processing Systems*, pp. 1269–1277 (2014)
 24. Chang, C.H., Kehtarnavaz, N.: Computationally efficient image deblurring using low rank image approximation and its GPU implementation. *J. Real Time Image Process.* **12**(3), 567–573 (2016)
 25. 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 (2017). arXiv preprint. [arXiv:1704.04861](https://arxiv.org/abs/1704.04861)
 26. Iandola, F.N., Han, S., Moskewicz, M.W., Ashraf, K., Dally, W.J., Keutzer, K.: SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5 mb model size (2016). arXiv preprint. [arXiv:1602.07360](https://arxiv.org/abs/1602.07360)
 27. Li, H., Kadav, A., Durdanovic, I., Samet, H., Graf, H.P.: Pruning filters for efficient convNets. In: *ICLR*, pp. 1–13 (2017)
 28. He, Y., Zhang, X., Sun, J.: Channel pruning for accelerating very deep neural networks. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1389–1397 (2017)
 29. 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/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4340–4349 (2019)
 30. Hu, H., Peng, R., Tai, Y.W., Tang, C.K.: Network trimming: a data-driven neuron pruning approach towards efficient deep architectures (2016). arXiv preprint. [arXiv:1607.03250](https://arxiv.org/abs/1607.03250)
 31. Lin, M., Ji, R., Wang, Y., Zhang, Y., Zhang, B., Tian, Y., Shao, L.: Hrank: filter pruning using high-rank feature map. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1529–1538 (2020)
 32. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images. Technical report, University of Toronto (2009)
 33. 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. *Int. J. Comput. Vis.* **115**(3), 211–252 (2015)
 34. Huang, Z., Wang, N.: Data-driven sparse structure selection for deep neural networks. In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 304–320 (2018)
 35. 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*, pp. 5058–5066 (2017)
 36. 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*, pp. 9194–9203 (2018)
 37. 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/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2790–2799 (2019)
 38. Lin, M., Cao, L., Li, S., Ye, Q., Tian, Y., Liu, J., Tian, Q., Ji, R.: Filter sketch for network pruning. In: *IEEE Transactions on Neural Networks and Learning Systems* (2021)
 39. Tian, G., Chen, J., Zeng, X., Liu, Y.: Pruning by training: a novel deep neural network compression framework for image processing. *IEEE Signal Process. Lett.* **28**, 344–348 (2021)
 40. Molchanov, P., Mallya, A., Tyree, S., Frosio, I., Kautz, J.: Importance estimation for neural network pruning. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11264–11272 (2019)
 41. Lian, Y., Peng, P., Xu, W.: Filter pruning via separation of sparsity search and model training. *Neurocomputing* **462**, 185–194 (2021)
 42. Luo, J.H., Wu, J.: Autopruner: an end-to-end trainable filter pruning method for efficient deep model inference. *Pattern Recognit.* **107**, 107461 (2020)
- Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.
- Xiaozhou Xu** received her B.S. degree from Nanjing Tech University, China, in 2018. She is currently working toward the Ph.D. degree with the institute of Cyber Systems and Control, Department of Control Science and Engineering, Zhejiang University. Her latest research interests include model compression and acceleration of neural network.
- Jun Chen** received the M.S. degree in automation from the Zhejiang University, Hangzhou, China, in 2020. He is currently working toward the Ph.D. degree with the Institute of Cyber Systems and Control, Department of Control Science and Engineering, Zhejiang University, Hangzhou, China. His research interests include neural network quantization and deep learning.
- Hongye Su** received the Ph.D. degree from Zhejiang University in 1995. He was a Professor with the Institute of Advanced Control, Zhejiang University, in 2000, and as the Deputy Director of the Institute of Advanced Control from 1999 to 2008. He is currently the Deputy Director of the Institute of Cyber-Systems and Control, Zhejiang University. He has authored/co-authored over 100 journal papers and four books. He has undertaken more than 10 national and provincial funds and state key projects as the principal investigator, including the National Outstanding Youth Science Foundation of China, the National Natural Science Foundation of China, the Teaching and research Award Program for Outstanding Young Teachers in Higher Education Institutions of OE, P.R.C., and projects for National High Technology Research and Development Program of China (863 Program). His research interests include process control and optimization theory and application.
- Lei Xie** received the B.S. and Ph.D. degrees from Zhejiang University, China, in 2000 and 2005, respectively. He is currently a Full Professor at the College of Control Science and Engineering, Zhejiang University. His current research activities culminated in over 50 articles that are published in internationally renowned journals and conferences, three book chapters and a book in the areas of process control, model predictive control, and applied multivariate statistics. His research interests focus on the interdisciplinary area of statistics and cyber system control theory.