Filter Pruning via Learned Representation Median in the Frequency Domain

Xin Zhang, Weiying Xie, *Member, IEEE,* Yunsong Li, *Member, IEEE,* Jie Lei, *Member, IEEE,* and Qian Du, *Fellow, IEEE*

***Abstract*—In this paper, we propose a novel filter pruning method for deep learning networks by calculating the Learned Representation Median in Frequency domain (LRMF). In contrast to the existing filter pruning methods that remove relatively unimportant filters in the spatial domain, our newly proposed approach emphasizes the removal of absolutely unimportant filters in the frequency domain. Through extensive experiments, we observed that the criterion for "relative unimportance" cannot be generalized well and that the discrete cosine transform (DCT) domain can eliminate redundancy and emphasize low-frequency representation, which is consistent with the human visual system. Based on these important observations, our LRMF calculates the learned representation median in the frequency domain and removes its corresponding filter, since it is absolutely unimportant at each layer. Thanks to this, the time-consuming fine-tuning process is not required in LRMF. Results show that LRMF outperforms state-of-the-art pruning methods. For example, with ResNet110 on CIFAR-10, it achieves a 52.3% FLOPs reduction with an improvement of 0.04% in Top-1 accuracy. With VGG16 on CIFAR-100, it reduces FLOPs by 35.9% while increasing accuracy by 0.5%. On ImageNet, ResNet18 and ResNet50 are accelerated by 53.3% and 52.7% with only 1.76% and 0.8% accuracy loss, respectively. The code is based on PyTorch and is available at https://github.com/zhangxin-xd/LRMF.**

***Index Terms*—Absolutely unimportant filter pruning, learned representation median, frequency domain transform.**

I. Introduction

GREAT strides have been made towards more precise computer vision tasks [1], [2], [3], [4], [5], [6], thanks in part to the deeper and wider architectures of CNNs. However, they come at the cost of heavy computational burdens and make it hard to be established on real-world applications. For example, ResNet152 [7] of 60.2M parameters occupies 231M storage space and needs more than 6 seconds to process an image (FLOPs = 11.3B). [8] claims VGGNet with 321.1M memory occupation requires 236mW power for a batch of three frames and 1.4s to deal with a frame. The storage, computational efficiency, and power consumption are crucial enabling factors for deployment on mobile and edge devices and some immediate web services. Under such circumstances, network compression techniques emerge as promising solutions, including compact network design [9], [10], low-rank decomposition [11], [12], knowledge distillation [13], [14], parameter quantization [15], [16], network pruning [17], [18], etc.

Among them, network pruning tackles the problem of finding unnecessary parameters and removing them in order to reduce computation and memory requirements of deep neural networks while keeping or even improving learning effectiveness [19]. Recent techniques on network pruning [20] can be further roughly subdivided into two categories: weight pruning [21], [22], [23], [24] and filter pruning [25], [26], [27], [28]. The most famous weight pruning algorithm is to obtain sparse weight matrices, but the resulting irregular structure limits the application on Basic Linear Algebra Subprograms (BLAS) libraries [29]. In contrast to irregular structures of weight pruning, filter pruning removes the entire filter.

The cornerstone theory of existing filter pruning methods is to remove less important filters depending on norm [30] or rank [11] in the spatial domain. **On one hand**, the so-called "less important" is actually a relative term. Fig. 1 demonstrates the output representations of pruned filters and their rank values. It shows that these filters can still dig out characteristics of inputs, such as outlines of objects. In contrast, the preserved filters with high-rank values cannot mine the underlying characteristics of inputs and seem to be redundant. Consequently, pruning filters using the "less important" criterion cannot approach the performance of baseline neural networks. **On the other hand**, the human visual system is sensitive to low-frequency representations [31], [32], but pixels in the spatial domain may be insufficiently prominent for these key components, as shown in Fig. 3. In contrast, the discrete cosine transform (DCT) domain emphasizes low-frequency representations, as shown in Fig. 3 (middle), which is consistent with the human visual system. Based on the DCT coefficients in Fig. 3, we also make another observation that most of the high-frequency components have zero coefficients and only a small fraction of coefficients are non-zero. This is because images have high spatial correlation, and slow variation in local regions corresponds to low-frequency components only.

In this study, we are motivated to approach the performance of baseline neural networks while reducing computation and memory requirements by pruning ***absolutely unimportant*** filters rather than ***relatively unimportant*** ones **in the frequency domain**. Different from the existing methods, we transform the learned representations into the frequency domain to

This work was supported in part by the National Natural Science Foundation of China under Grant 62121001, Grant 62071360, Grant 61801359, and the Fundamental Research Funds for the Central Universities and the Innovation Fund of Xidian University under Grant 5001-20109215456.

X. Zhang, W. Xie, Y. Li, and J. Lei are with the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China (e-mail: xinzhang_xd@163.com; wyxie@xidian.edu.cn; ysli@mail.xidian.edu.cn.)

Q. Du is with the Department of Electronic and Computer Engineering, Mississippi State University, Starkville, MS 39759 USA (e-mail: du@ece.msstate.edu).

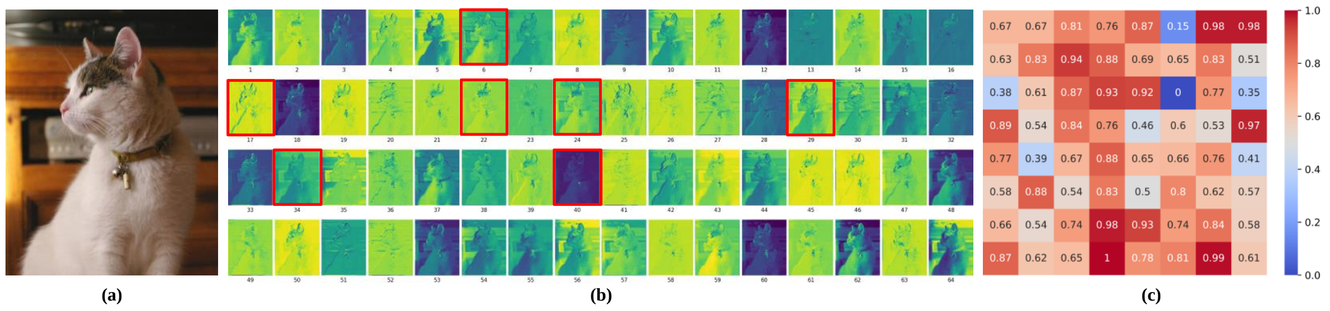


Fig. 1. Input image (a), visualization of learned representations (b) and heatmap of rank (c). The filters corresponding to the learned representations with red bounding boxes (6, 17, 22, 24, 29, 34, 40) are deactivated according to rank when the pruning rate is 0.12. These representations imply some distinctive and significant details, such as outlines of the bell and the cat's head, while some representations with high rank (8, 19, 23, 27, 36, 45, 55) provide a valid description of the input and seem to be redundant.

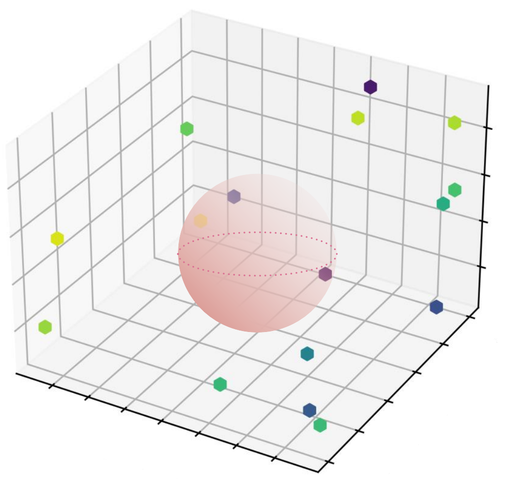


Fig. 2. The illustration of Representation Median (RM) in the frequency domain. The color dots are the representations in the frequency domain. The three dots in the sphere have the smallest sums of distances to other dots which suggests they are medians of representations and can be replaced by others. As the size of the sphere changes, the number of medians, that is, the number of pruning filters, changes.

find absolutely unimportant ones. For the powerful energy compaction capability of the DCT[1] (see Fig. 3), we leverage it to reveal sensitive components of the learned representations. In particular, since **Representation Median** (RM) in the frequency domain can be replaced by its surrounding representations and impact less on performance (see Fig. 2), it is removed at each layer without degrading performance. The proposed pruning mechanism, named LRMF (Learned Representation Median in the Frequency domain), is shown in Fig. 4. It can achieve 93.88% Top-1 accuracy with 52.3% FLOPs reduced when pruning ResNet110 on CIFAR-10, exceeding existing filter pruning methods. Notably, the Top-1 accuracy of VGG16 pruned with LRMF exceeds the baseline, improving performance by 0.5% and reducing FLOPs by 35.9% on CIFAR-100. On ImageNet, ResNet18 and ResNet50 are accelerated by 53.3% and 52.7% with only 1.76% and 0.8% accuracy loss, respectively. More significantly, LRMF can be well generalized to different datasets and does not require fine-tuning.

Compared with existing filter pruning algorithms, the main contributions of the proposed LRMF method lie in the followings.

- The concept of absolutely unimportant representations is introduced into filter pruning, which not only approaches the performance of the baseline with FLOPs reduction but also opens up more flexible ways for pruning.
- To satisfy the concept of absolutely unimportant representations, Representation Median (RM) in the frequency domain is calculated according to the output representations within the same layer, which takes into account both the input images and the filter attributes.
- Our LRMF provides a novel perspective for structured pruning. The DCT conducted on the learned representations which are images instead of treating filters as images is consistent with the sensitivity of the human visual system to low-frequency representations and allows for the elimination of redundancy and correlation in the spatial domain which enables the computation reduction in the pruning stage.
- The extensive experiments performed on several benchmarks using both multiple-branch network and single-branch network show that the proposed LRMF model boosts the state-of-the-art pruning performance in terms of effectiveness, efficiency, and generalization.

II. Related Work

Network compression. The formidable amount of computational resources used by CNNs present a great challenge in its deployment in cost-sensitive cloud services and low-powered edge-computing applications. To address this problem, many efforts have been made in network compression, which can be divided into five categories, *i.e.*, compact network design, low-rank decomposition, knowledge distillation, parameter quantization, and network pruning. ***Compact network design*** is the most direct method to adapt the resource-hungry nature. InceptionNet [33] aggregates several receptive fields of different sizes, improving network accuracy, simultaneously, resorts to 1×1 convolution to keep the computational complexity small. Pointwise group convolution and channel

[1]In this paper, we use the terms frequency domain and DCT domain interchangeably.

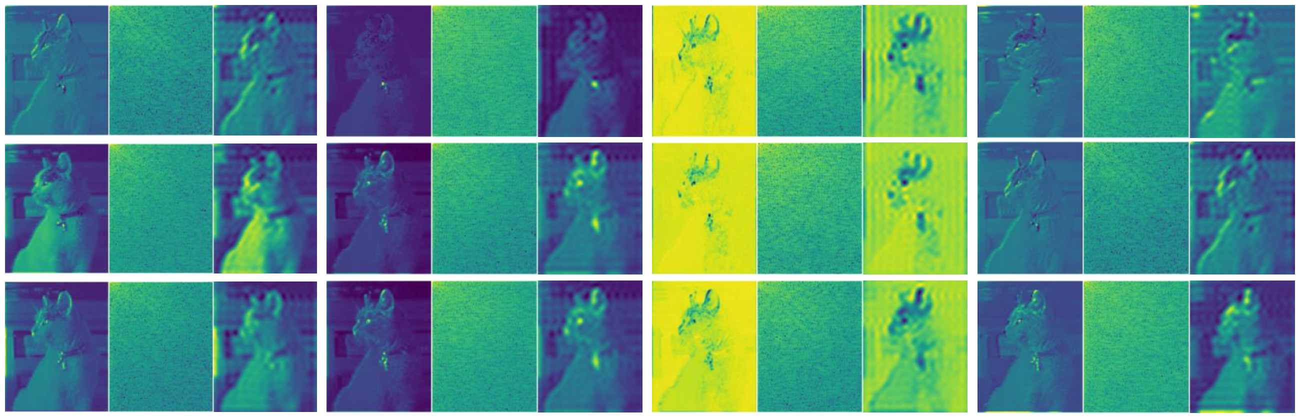


Fig. 3. Learned representations (left), DCT results (middle), and low-frequency representations (right). Similar learned representations in spatial domain can be better distinguished in the low-frequency domain.

shuffle are the corn of ShuffleNet [34] and other compact networks such as MobileNet [9], SqueezeNet [35], and recent GhostNet [10] are constructed based on depthwise separable convolution, fire module, and ghost module, respectively. ***Low-rank decomposition*** roots in matrix algebra and achieves network compression by approximating each layer's weight matrix by a low-rank matrix, which can be tackled with singular value decomposition (SVD). [11] solved the fundamental problem of rank selection with a mixed discrete-continuous optimization. Instead of considering only linear filters or linear responses, [12] minimizes the reconstruction error of nonlinear responses, subject to a low-rank constraint, which helps reduce the complexity of filters. [36] resorts to a low-rank decomposition to deal with the compression in the last convolution layer in a ResBlock in a global manner. ***Knowledge distillation (KD)*** transfers knowledge from a pre-trained large "teacher-net" (or even an ensemble of networks) to a small "student-net," for facilitating the deployment at test time. Typically, the original network and the compressed network are treated as "teacher-net" and "student-net", respectively, and knowledge transfer is executed with various kinds of matching mechanisms. [37] designs an adversarial game between "teacher-net" and "student-net" by adding a discriminator. [38] uses KD to compensate for the weakness of limited data. [14] proposed a novel alignment method by adding a 1×1 convolution layer at the end of each layer block of the student-net, respectively, and fitting the block-level outputs of the "student-net" to the "teacher-net" by estimating the parameters of the added layers. ***Parameter quantization*** encodes the values of weights in a low precision way. Generally speaking, float (fp32) is used as the data representation in the construction and training of neural networks. Parameter quantization uses half-precision floating-point (fp16), 8-bit fixed-point integer (int8 / uint8), and other low precision numerical formats for storage and calculation. [15] designs a baseline binary network by modifying MobileNetV1 and uses a simple channel-wise reshaping and shifting operation and a distributional loss between binary and real-valued network outputs to strike a good tradeoff between accuracy and efficiency. ***Network pruning*** is detailed below.

Network pruning. Network pruning is a predominant approach for removing redundancy in CNNs which can be weight pruning [21], [22] or filter pruning [25], [26]. In virtue of its better compatibility with off-the-shelf computing libraries, filter pruning is preferred and is illustrated in Fig. 5.

Filter pruning can establish a compact neural network by eliminating unnecessary convolutional filters. The primary issue raised in filter pruning concerns how to establish a reasonable guideline to find the filters that should be pruned. In this respect, several recent studies involve the concept of relativity, and among them, norm [30], [39], energy [40], and rank [25], [11] are about the removal of relatively unimportant filters. In addition to the above-mentioned concept, in FPGM [29], the filters with redundant information are removed, which is not sensitive to the adoption of the "smaller-norm-less-important" criterion [41]. Obviously, these works implement filter pruning in the spatial domain. Unfortunately, based on the observation in Fig. 3, pixels in the spatial domain may be insufficiently prominent and redundant for the key low-frequency representations of interest to the human visual system.

Frequency Domain Learning. As a representative work, Gueguen *et al.*[42] directly inputs the DCT coefficients into the ResNet [7]. Ehrlich *et al.*[43] expresses deep residual learning in the JPEG domain. These previous works validate that learning in the frequency domain converges faster. In [44], [45], and [46], the authors conduct extensive experiments to demonstrate that learning in the frequency domain can reduce the sensitivity to different datasets thus extending model generalization. Recently, Xu *et al.*[31] shows that learning in the frequency domain can improve the performance of instance segmentation, which also validates that deep models are more sensitive to low-frequency channels. Inspired by the above-mentioned advances in frequency domain-based learning, our proposal removes filters according to absolutely unimportant representations in the frequency domain. There are some prior works in frequency domain-based pruning. For instance, [47] transforms the convolution in spatial domain into the product in frequency domain to remove the spatial redundancy within most filters. By treating convolution filters as images, [48] aims for weight pruning and decomposes their

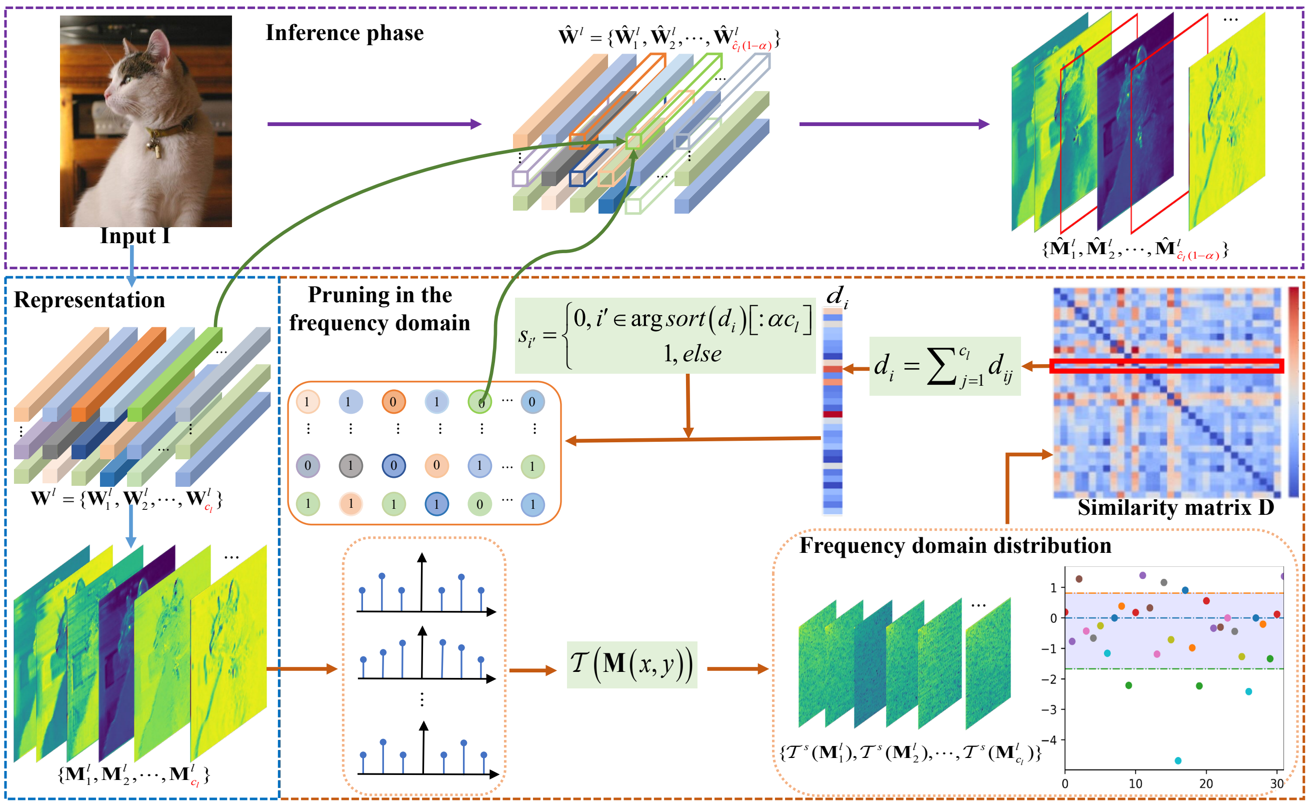

Inference phase
Input I
Representation
Pruning in the frequency domain
Similarity matrix D
Frequency domain distribution
d_i
$d_i = \sum_{j=1}^{c_l} d_{ij}$
$s_{i'} = \begin{cases} 0, i' \in \arg sort(d_i)[:\alpha c_l] \\ 1, else \end{cases}$
$\mathcal{T}(\mathbf{M}(x,y))$
$\mathbf{W}^l = \{\mathbf{W}_1^l, \mathbf{W}_2^l, \cdots, \mathbf{W}_{c_l}^l\}$
$\{\mathbf{M}_1^l, \mathbf{M}_2^l, \cdots, \mathbf{M}_{c_l}^l\}$
$\hat{\mathbf{W}}^l = \{\hat{\mathbf{W}}_1^l, \hat{\mathbf{W}}_2^l, \cdots, \hat{\mathbf{W}}_{\hat{c}_l(1-\alpha)}^l\}$
$\{\hat{\mathbf{M}}_1^l, \hat{\mathbf{M}}_2^l, \cdots, \hat{\mathbf{M}}_{\hat{c}_l(1-\alpha)}^l\}$
$\{\mathcal{T}^s(\mathbf{M}_1^l), \mathcal{T}^s(\mathbf{M}_2^l), \cdots, \mathcal{T}^s(\mathbf{M}_{c_l}^l)\}$

Fig. 4. Framework of LRMF. First, representations are generated when an image runs through the convolution layers. Then, we remove filters in the frequency domain.

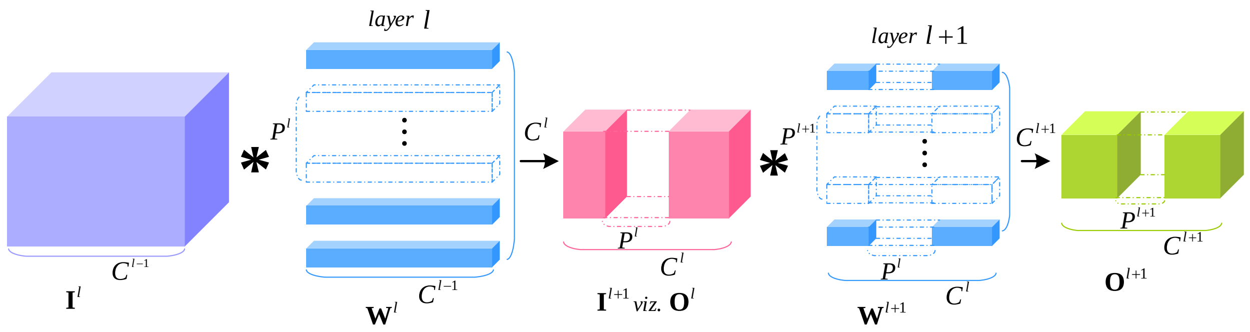


Fig. 5. The illustration of filter pruning. $\mathbf{I}^l$ and $\mathbf{I}^{l+1}$ are the input of the lth layer and $l+1$th layer with C^l and $C^{l+1}-P^l$ channels, respectively. $\mathbf{O}^l$ and $\mathbf{O}^{l+1}$ are the output of the lth layer and $l+1$th layer with $C^{l+1}-P^l$ and $C^{l+2}-P^{l+1}$ channels, respectively. $\mathbf{W}^l$ and $\mathbf{W}^{l+1}$ are the weights of the lth layer and $l+1$th layer, containing C^{l+1} and C^{l+2} filters and corresponding to the input of the current layer in size. Filter pruning in each convolution layer leads to channel reduction in the next layer. P^l and P^{l+1} filters in the lth layer and $l+1$th layer are trimmed off, which result in channel reduction of P^l and P^{l+1}.

representations in the frequency domain as common parts and individual private parts. Different from these methods, our LRMF specially formulates for structured pruning, and conducts a more rational frequency domain transformation on the learned representations which are images instead of treating filters as images.

III. Methodology

A. Problem Formulation

For an L-layer baseline neural network $\mathcal{N}$, let $\mathcal{C}=(c_1, c_2, \ldots, c_L)$ denote its convolutional layer, where c_l is the number of filters in the lth layer. Let $\mathbf{W}^l = \{\mathbf{W}_1^l, \mathbf{W}_2^l, \cdots, \mathbf{W}_{c_l}^l\}$ be the parameters in the lth layer, where $\mathbf{W}_k^l$ represents the kth parameters in the lth layer. The aim of the proposed LRMF is to find an optimal characterization subset of $\mathcal{N}$ by removing redundant and replaceable filters in order to reduce computational burdens. The objective function can be formulated as:

$$\begin{gathered} \max_{s_{l,k}} \sum_{l=1}^{L} \sum_{k=1}^{c_l} s_{l,k} \mathcal{R}\left(\mathbf{W}_k^l\right) \\ s.t. \sum_{k=1}^{c_l} s_{l,k} = (1-\alpha)\, c_l \end{gathered}, \tag{1}$$

where $s_{l,k}$ is the gate value for the kth filter in the lth layer, which is either 0 or 1, with 0 being pruned and 1 being preserved. $\mathcal{R}(\cdot)$ evaluates the representation capability of filter $\mathbf{W}_k^l$. The constraint refers to the number of filters retained in the lth layer, where α is the compression ratio, which is a constant. See Section IV-B for details. Note that the core is to determine the representation function $\mathcal{R}(\cdot)$ and the gate value s for each filter. We start by redefining $\mathcal{R}(\cdot)$ to make it more in line with the human visual system.

Inspired by the recent advances in filter pruning design, we define $\mathcal{R}(\cdot)$ on the learned representations because they can characterize both input images and filter attributes. Let $\mathbf{I}^l = \{\mathbf{I}_1^l, \mathbf{I}_2^l, \cdots, \mathbf{I}_{c_{l-1}}^l\}$ denote the input at the lth layer, where $\mathbf{I}_k^l$ represents the kth input at the lth layer. Eq. (1) can be reformulated as:

$$\begin{aligned} \max_{s_{l,k}} \sum_{l=1}^{L}\sum_{k=1}^{c_l} s_{l,k}\mathcal{R}\left(\mathbf{I}^l * \mathbf{W}_k^l\right) \\ s.t. \sum_{k=1}^{c_l} s_{l,k} = (1-\alpha)\, c_l \end{aligned}, \tag{2}$$

where $*$ denotes the convolution operation. Let $\mathbf{M} = \mathbf{I}*\mathbf{W}$ for better presentation. Here, we omit the activation and bias for simplicity. To solve Eq. (2), we should design a reasonable function $\mathcal{R}(\cdot)$ by which the pruned filters are absolutely unimportant, ensuring that the preserved filters maximize the characterization of the baseline neural network $\mathcal{N}$. Based on the fact that the RM can be replaced by its surrounding representations and contributes less to performance, we preserve the representation with the least similarity to others, *i.e.*, the representation with the greatest sum of distances from other representations. Consequently, we design our representation function $\mathcal{R}(\cdot)$ as:

$$\mathcal{R}\left(\mathbf{M}_k^l\right) = \sum_{i=1}^{c_l} \left\|\mathbf{M}_k^l - \mathbf{M}_i^l\right\|_2, \tag{3}$$

which is to calculate similarity in the form of Euclidean distance. Here, $\mathbf{M}_k^l$ represents the kth out representation in the lth layer.

B. Solving in the Frequency Domain

The findings in Fig. 3 motivate us to transform the computation of Eq. (3) in the spatial domain into the frequency domain to eliminate trivial frequency components and reduce the input size, consequently decreasing computational burdens. Besides, the learned representations in the frequency domain further emphasize the "absolute unimportant" perspective since only a small fraction of frequency coefficients are non-zero, while the majority are zero because of redundancy and correlation in the spatial domain. Based on the aforementioned discussion and the importance of low-frequency characteristics, we perform a simple yet effective operation as

$$\mathcal{R}\left(\mathcal{T}\left(\mathbf{M}_k^l\right)\right) = \sum_{i=1}^{c_l} \left\|\mathcal{T}\left(\mathbf{M}_k^l(x,y)\right) - \mathcal{T}\left(\mathbf{M}_i^l(x,y)\right)\right\|_2, \tag{4}$$

where (x,y) represents the position of a pixel in the spatial domain, and $\mathcal{T}(\cdot)$ denotes the DCT operation, which has the specific form of

$$\begin{aligned} \mathcal{T}(\mathbf{M}(x,y)) = \varepsilon(u)\,\varepsilon(v) \sum_{x=0}^{h-1}\sum_{y=0}^{w-1} \mathbf{M}(x,y) \\ \cos\left[\frac{(x+0.5)\,\pi}{Z}\right]\cos\left[\frac{(y+0.5)\,\pi}{Z}\right], \end{aligned} \tag{5}$$

with (u,v) being the position of a pixel in the DCT domain. For convenience, we use $\mathbf{M}$ to denote $\mathbf{M}_k^l$, h and w represent the height and width of the learned representation, respectively, and $Z = h \times w$. Generally, $\varepsilon(u)$ is defined as

$$\varepsilon(u) = \begin{cases} \sqrt{1/Z}, u = 0 \\ \sqrt{2/Z}, u \neq 0 \end{cases}. \tag{6}$$

Likewise for $\varepsilon(v)$. Due to the powerful energy compaction capability of the DCT, as expected, the sensitive components of $\mathbf{M}$ are concentrated in the upper left corner of $\mathcal{T}(\mathbf{M})$ (see Fig. 4). It is denoted by $\mathcal{T}^s(\mathbf{M})$ and is smaller in size than $\mathbf{M}$. Eq. (4) is further formulated as

$$\mathcal{R}\left(\mathcal{T}^s\left(\mathbf{M}_k^l\right)\right) = \sum_{i=1}^{c_l} \left\|\mathcal{T}^s\left(\mathbf{M}_k^l\right) - \mathcal{T}^s\left(\mathbf{M}_i^l\right)\right\|_2. \tag{7}$$

C. Pruning Stage

Note that the calculation quantities of Eq. (3) and Eq. (7) are $\sum_{l=1}^{L} l \times c_l^2 \times h \times w$ and $\sum_{l=1}^{L} l \times c_l^2 \times a \times b$, respectively. Let a and b represent the height and width of $\mathcal{T}^s(\mathbf{M})$, respectively. In our settings, a is $\frac{1}{4}h$ and b is $\frac{1}{4}w$. It is intuitive that our pruning mechanism in the frequency domain can reduce computational burdens. To solve Eq. (7), a similarity matrix is introduced

$$\mathbf{D} = \begin{pmatrix} d_{11} & d_{12} & \cdots & d_{1c_l} \\ d_{21} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ d_{c_l1} & d_{c_l2} & \cdots & d_{c_lc_l} \end{pmatrix}, \tag{8}$$

where d_{ij} represents the distance between $\mathcal{T}^s\left(\mathbf{M}_i^l\right)$ and

Algorithm 1 Algorithm Description of LRMF

```
Input: Training data I, baseline neural network $\mathcal{N}$, compression ratio $\alpha$, prune interval
Output: The pruned model $\mathcal{N}^*$
1:  Initialize $\mathcal{N}$ with pretrained or default initialization parameters;
2:  for each epoch do
3:      Optimize the model with SGD;
4:      if epoch % prune interval==0 then
5:          for $l$ in range $L$ do
6:              Transform representations using DCT with Eq. (5);
7:              Calculate similarity matrix in Eq. (8) with Eq. (7);
8:              Choose $\alpha c_l$ filters according to Eq. (9) to prune;
9:          Update the model with pruned $\mathcal{N}^*$;
10:     Save the best model according to the performance on validation set;
11: return The best pruned model $\mathcal{N}^*$
```

TABLE I
PRUNING RESULTS OF RESNET AND VGGNET ON CIFAR-10

Network	Fine-tune?	Method	Top-1(%)	ΔTop-1(%)↓	FLOPs	ΔFLOPs(%)↓
ResNet20	✗	SFP[39]	$92.20 \rightarrow 90.83$	1.37	**24.3M**	**42.2**
	✗	LRMF($\alpha = 0.3$)	**$92.20 \rightarrow 91.04$**	**0.16**	**24.3M**	**42.2**
	✗	FPGM[29]	$92.20 \rightarrow 90.44$	1.76	**18.7M**	**54.0**
	✗	LRMF($\alpha = 0.4$)	**$92.20 \rightarrow 90.47$**	**1.73**	**18.7M**	**54.0**
ResNet32	✗	MIL[49]	$92.33 \rightarrow 90.47$	1.59	47.0M	31.2
	✗	SFP[39]	**$92.63 \rightarrow 92.08$**	**0.55**	**40.3M**	41.5
	✗	FPGM[29]	$92.63 \rightarrow 91.93$	0.7	**32.3M**	**53.2**
	✗	LRMF($\alpha = 0.4$)	**$92.63 \rightarrow 92.08$**	**0.55**	**32.3M**	**53.2**
	✓	TAS[50]	-	0.73	35.0M	49.4
	✓	LRMF($\alpha = 0.4$)	**$92.63 \rightarrow 92.08$**	**0.55**	**32.3M**	**53.2**
ResNet56	✗	PFEC[30]	$93.04 \rightarrow 91.31$	1.75	90.9M	27.6
	✗	LSTM[51]	-	0.87	-	47.5
	✗	CP[52]	$92.80 \rightarrow 90.90$	1.90	-	50.0
	✗	SFP[39]	$93.59 \rightarrow 92.26$	1.33	**59.4M**	**52.6**
	✗	Rethink[53]	$93.80 \rightarrow 92.80$	1.00	**59.4M**	**52.6**
	✗	ASFP[54]	$93.59 \rightarrow 92.44$	1.15	**59.4M**	**52.6**
	✗	FPGM[29]	$93.59 \rightarrow 92.93$	0.66	**59.4M**	**52.6**
	✗	LRMF($\alpha = 0.4$)	**$93.59 \rightarrow 93.29$**	**0.30**	**59.4M**	**52.6**
	✓	CP[52]	$93.26 \rightarrow 90.80$	2.46	62.0M	50.6
	✓	DCP[55]	$93.80 \rightarrow 93.50$	0.30	62.7M	50.0
	✓	ASFP[54]	$93.59 \rightarrow 93.12$	0.47	**59.4M**	**52.6**
	✓	TAS[50]	-	0.77	**59.5M**	**52.7**
	✓	LRMF($\alpha = 0.4$)	**$93.59 \rightarrow 93.25$**	**0.34**	**59.4M**	**52.6**
ResNet110	✗	MIL[49]	$93.63 \rightarrow 93.44$	0.19	-	34.2
	✗	PFEC[30]	$93.53 \rightarrow 92.94$	0.61	155M	38.6
	✗	SFP[39]	$93.68 \rightarrow 93.38$	0.30	150M	40.8
	✗	Rethink[53]	$93.77 \rightarrow 93.70$	0.07	150M	40.8
	✗	FPGM[29]	**$93.68 \rightarrow 93.73$**	**-0.05**	**121M**	**52.3**
	✗	LRMF($\alpha = 0.4$)	**$93.68 \rightarrow 93.72$**	**-0.04**	**121M**	**52.3**
	✓	PFEC[30]	$93.53 \rightarrow 93.30$	0.23	155M	38.6
	✓	GAL[56]	$93.26 \rightarrow 92.74$	0.81	-	40.5
	✓	Rethink[53]	$93.77 \rightarrow 93.89$	-0.12	150M	40.8
	✓	NISP[57]	-	0.18	-	43.8
	✓	FPGM[29]	$93.68 \rightarrow 93.74$	-0.06	**121M**	**52.3**
	✓	SFP[39]	$93.68 \rightarrow 92.90$	0.78	**121M**	**52.3**
	✓	LRMF($\alpha = 0.4$)	**$93.68 \rightarrow 93.88$**	**-0.20**	**121M**	**52.3**
	✓	TAS[50]	-	0.64	119M	53.0
	✓	HRank[25]	$93.50 \rightarrow 93.36$	0.14	105.7M	58.2
	✓	LFPC[58]	$93.68 \rightarrow 93.07$	0.61	101M	60.3
	✓	LRMF($\alpha = 0.5$)	**$93.68 \rightarrow 93.61$**	**0.07**	**94.0M**	**62.8**
VGG16	✗	FPGM[29]	**$93.58 \rightarrow 93.23$**	**0.35**	-	**35.9**
	✗	LRMF($\alpha = 0.2$)	**$93.58 \rightarrow 93.20$**	**0.38**	-	**35.9**
	✓	PFEC[30]	$93.58 \rightarrow 93.28$	0.30	-	34.2
	✓	LRMF($\alpha = 0.2$)	**$93.58 \rightarrow 93.70$**	**-0.12**	-	**35.9**

Note: "✗" indicates pruning without fine-tuning, and "✓" indicates with fine-tuning. "↓" indicates the difference between pruned and baseline neural networks.

$\mathcal{T}^s\left(\mathbf{M}_j^l\right)$, *i.e.*, $d_{ij} = \left\|\mathcal{T}^s\left(\mathbf{M}_i^l\right) - \mathcal{T}^s\left(\mathbf{M}_j^l\right)\right\|_2$. $\mathbf{D}$ is a symmetric matrix and its diagonal entry is 0. d_i corresponds to the summation of all the similarities related to the ith learned representation in the frequency domain, *i.e.*, $d_i = \sum_{j=1}^{c_l} d_{ij}$. It is worth noting that the larger d_i is, the more unique and irreplaceable information is contained in the ith learned representation, which means its corresponding filter should be preserved to approach the characterization of the baseline neural network. Based on this analysis, the gate s_i concretely satisfies the following constraint:

$$s_{i'} = \begin{cases} 0, & i' \in \arg sort\left(d_i\right)[: \alpha c_l] \\ 1, & else \end{cases} \tag{9}$$

Here, $sort\left(d_i\right)$ means to sort d_i in ascending order, and αc_l indicates the number of filters pruned at the lth layer. Therefore, αc_l filters corresponding to the learned representations with smaller d_i are pruned. Combining Eq. (1), Eq. (7), and Eq. (9), the final objective of LRMF gives:

$$\max_{l,k,i} \sum_{l=1}^{L} \sum_{k=1}^{c_L} \sum_{i \notin A}^{c_l} \left\| \mathcal{T}^s\left(\mathbf{M}_k^l\right) - \mathcal{T}^s\left(\mathbf{M}_i^l\right) \right\|_2 . \tag{10}$$
$$s.t.\ A = \{i | i \in \arg sort\,(d_i)\,[: \alpha c_l]\}$$

Eq. (10) is optimized in the training phase and the model with the best performance on validation set is saved as the final pruned model $\mathcal{N}^*$ which has a structure of $\hat{\mathcal{C}} = (\hat{c_1}, \hat{c_2}, \ldots, \hat{c_L})$. The parameters in the lth layer of $\mathcal{N}^*$ can be denoted as $\hat{\mathbf{W}}^l = \{\hat{\mathbf{W}}_1^l, \hat{\mathbf{W}}_2^l, \cdots, \hat{\mathbf{W}}_{\hat{c}_l(1-\alpha)}^l\}$.

The pseudo-code for our LRMF is summarized in Algorithm 1, where "%" indicates taking the remainder. First, we obtain the learned representations $\mathbf{M}$ by inputting $\mathbf{I}$ into a baseline neural network $\mathcal{N}$. Second, we transform $\mathbf{M}$ into the frequency domain using DCT resulting in $\mathcal{T}(\mathbf{M})$. Third, we calculate the similarity matrix $\mathbf{D}$ among the sensitive components of $\mathcal{T}(\mathbf{M})$, *i.e.*, $\mathcal{T}^s(\mathbf{M})$. Fourth, we sort d_i corresponding to the summation of all the similarities related to the ith $\mathcal{T}^s(\mathbf{M})$ in ascending order. As shown in Fig. 4, we finally remove αc_l filters at each layer based on Eq. (10) arriving at the pruned network $\mathcal{N}^*$. It should be emphasized that the time-consuming fine-tuning process is not employed in LRMF.

IV. Experiments

A. Experimental Settings

Datasets and Baseline. We conduct extensive experiments on three popular datasets, *i.e.*, CIFAR-10 [59], CIFAR-100 [59], and ImageNet [60], to investigate LRMF for representative networks, including multiple-branch network (ResNet) [7] and single-branch network (VGGNet) [61]. We compare LRMF with 17 state-of-the-art or often-cited methods, including SFP [39][2], FPGM [29][3], MIL [49], TAS [50][4], PFEC [30][5], LSTM [51], CP [52][6], Rethink [53][7], Asymptotic SFP (ASFP) [54], DCP [55][8], GAL [56][9], NISP [57], HRank [25][10], LFPC [58], CFP [62], GDP [63], and SCP [64].

Performance Metric. FLOPs is employed to evaluate computational complexity. With the output feature map $\mathbf{I}^{l+1} \in \mathbb{R}^{h_{l+1} \times w_{l+1} \times c_l}$ and the weight $\mathbf{W}^l \in \mathbb{R}^{k_l \times k_l \times c_{l-1} \times c_l}$ of the lth layer, FLOPs is caculated with

$$\text{Flops} = \sum_{l=1}^{L} h_{l+1} \times w_{l+1} \times c_{l-1} \times c_l \times k_l \times k_l \quad , \tag{11}$$

where c_{l-1} is the channel number of the input feature map and c_l is the number of filters in the lth layer.

As for accuracy evaluatment, we demonstrate Top-1 accuracy of the pruned models for CIFAR-10 and CIFAR-100 and report both Top-1 and Top-5 accuracies on ImageNet.

[2]https://github.com/he-y/soft-filter-pruning
[3]https://github.com/he-y/filter-pruning-geometric-median
[4]https://github.com/D-X-Y/NAS-Projects
[5]https://github.com/Eric-mingjie/rethinking-network-pruning/tree/master/imagenet/l1-norm-pruning
[6]https://github.com/yihui-he/channel-pruning
[7]https://github.com/Eric-mingjie/rethinking-network-pruning
[8]https://github.com/SCUT-AILab/DCP
[9]https://github.com/ShaohuiLin/GAL
[10]https://github.com/lmbxmu/HRank

TABLE II
Pruning Results of ResNet and VGGNet on CIFAR-100

Network	F.T	Method	ΔTop-1(%)↓	ΔFLOPs(%)↓
ResNet56	✗	MIL[49]	2.96	39.3
	✗	LFPC[58]	0.58	51.6
	✗	SFP[39]	2.61	**52.6**
	✗	FPGM[29]	1.75	**52.6**
	✗	LRMF($\alpha = 0.4$)	**0.44**	**52.6**
VGG16	✗	FPGM[29]	-0.20	**35.9**
	✗	LRMF($\alpha = 0.2$)	**-0.50**	**35.9**

Note: "F.T" means fine-tuning.

Configurations. The proposed LRMF method is implemented with Pytorch. All the networks are optimized by using Stochastic Gradient Descent (SGD) algorithm. The ResNet-20, 32, 56, 110 are trained for 300 epochs on CIFAR dataset. The momentum is 0.9, the weight decay is 0.005 and the batch size is 128. The initial learning rate is 0.01 and decays by 10 at Epoch 150 and 225. The training schedule of VGGNet on CIFAR dataset is configured as [29].

For ImageNet, ResNet18 and ResNet50 are trained for 100 epochs on 8 NVIDIA RTX 2080Ti GPUs with batch size of 256, weight decay of 1e-4, and momentum of 0.9.

Both starting from scratch and fine-tuning with pretrained model are considered. For the latter one, we reduce the learning rate to one-tenth of the original one.

B. Results and Analysis

1) Results on CIFAR-10: In Table I, we evaluate LRMF on CIFAR-10 and compare against prior art for popular CNNs including single-branch network (VGGNet) and multiple-branch network (ResNet).

ResNet. The proposed LRMF is tested on ResNet-20, 32, 56, 110. As shown in Table I, LRMF achieves the sate-of-the-art performance. For example, pruning from scratch, LRMF on ResNet20 achieves higher classification accuracy (91.04 v.s. 90.83 and 90.47 v.s. 90.44) while maintaining the same speedup ratio as SFP [39], and FPGM [29], when the compression ratio $\alpha = 0.3$ and $\alpha = 0.4$. In comparison with MIL [49], SFP [39], and FPGM [29], LRMF strikes a better balance between acceleration rate and accuracy on ResNet32. When fine-tune is true, LRMF achieves lower performance degradation with higher FLOPs reduction than TAS [50]. For pruning ResNet56 from scratch, LRMF obtains an excellent Top-1 accuracy with more significant FLOPs reduction than PFEC [30], LSTM [51], and CP [52]. Besides, we observe that LRMF shares the same speedup ratio as SFP [39], Rethink [53], ASFP [54], and FPGM [29], but yields a lower performance decrease (0.3% vs. 1.33%, 0.3% vs. 1.00%, 0.3% vs. 1.15%, and 0.3% vs. 0.55%). LRMF with fine-tuning accelerates ResNet56 by 52.6% speedup ratio with 0.34% accuracy drop which outperforms CP [52], ASFP [54], and TAS [50] is comparable to DCP [55]. Compared with MIL [49], PFEC [30], SFP [39] and Rethink [53] without fine-tuning, LRMF is advantageous in both Top-1 accuracy and FLOPs reduction on ResNet110. In addition, LRMF nearly approaches FPGM [29] that is an effective channel pruning

TABLE III
PRUNING RESULTS OF RESNET18 AND RESNET50 ON IMAGENET

Network	Fine-tune?	Method	Top-1(%)	ΔTop-1(%)↓	Top-5(%)	ΔTop-5(%)↓	ΔFLOPs(%)↓
ResNet18	✗	MIL[49]	$69.98 \rightarrow 66.33$	3.65	$89.24 \rightarrow 86.94$	2.30	34.6
	✗	SFP[39]	$70.28 \rightarrow 67.10$	3.18	$89.63 \rightarrow 87.78$	1.85	41.8
	✗	FPGM[29]	$70.28 \rightarrow 67.78$	2.50	$\mathbf{89.63 \rightarrow 88.01}$	**1.62**	41.8
	✗	LRMF($\alpha = 0.4$)	$\mathbf{70.28 \rightarrow 67.87}$	**2.41**	$\mathbf{89.63 \rightarrow 87.87}$	**1.76**	**53.3**
ResNet50	✗	SFP[39]	$76.15 \rightarrow 74.61$	1.54	$92.87 \rightarrow 92.06$	0.81	41.8
	✗	CFP[62]	-	1.90	-	**0.80**	49.6
	✗	CP[52]	-	-	-	1.40	50.0
	✗	GDP[63]	-	3.24	-	1.59	51.3
	✗	FPGM[29]	$76.15 \rightarrow 74.13$	2.02	$92.87 \rightarrow 91.94$	0.93	**53.5**
	✗	SCP[64]	-	1.69	-	0.98	**54.3**
	✗	LRMF($\alpha = 0.4$)	$\mathbf{76.15 \rightarrow 74.70}$	**1.45**	$\mathbf{92.87 \rightarrow 92.07}$	**0.80**	**52.7**

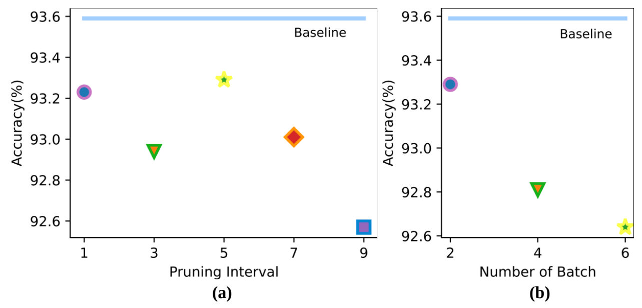


Fig. 6. Influence of pruning interval (a) and number of batch for DCT (b) on pruning.

method. For pruning pretrained ResNet110, LRMF retains comparable or even higher accuracies against the baseline neural network under two speedup ratios. Compared with other methods, LRMF shows absolute predominance. The superior performance on ResNet110 indicates that LRMF has a powerful ability to accelerate the deeper and more over-parameterized neural network.

VGGNet. Whether fine-tune or not, LRMF performs better than PFEC [30] in both accuracy and FLOPs reduction on VGG16. Besides, LRMF yields roughly the same performance (93.23 v.s. 93.20) as FPGM [29] without the fine-tuning. Moreover, for pruning the pre-trained VGG16, LRMF achieves a 35.9% acceleration ratio of the model and even improves the Top-1 accuracy from 93.58 to 93.70. These results demonstrate the potential of pruning in the frequency domain and the effectiveness of our LRMF design.

2) Results on CIFAR-100: Table II reports the pruning results of ResNet56 obtained by MIL [49], LFPC [58], SFP [39], FPGM [29], and our LRMF on the CIFAR-100 dataset. It can be observed that LRMF outperforms other compared methods. Concretely, LRMF reduces more than 50% FLOPs of ResNet56 with only 0.44% accuracy loss when compression ratio $\alpha = 0.4$. However, MIL accelerates the network by 39.3% with nearly 7 times performance degradation. Even the nearest rival, LFPC, loses 0.58% classification accuracy. In addition, the Top-1 accuracy of VGGNet pruned with LRMF exceeds the baseline, improving performance by 0.5%.

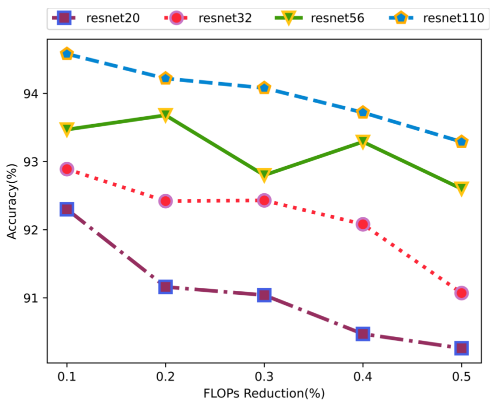
3) Results on ImageNet: We test ResNet18 and ResNet50 on ImageNet. Table III shows that LRMF outperforms the competing methods in both accuracy and FLOPs reduction. For ResNet18, LRMF reduces more than 50% FLOPs with the least Top-1 accuracy loss and an acceptable Top-5 accuracy loss. MIL has the lowest FLOPs reduction but the worst performance. LRMF is a performance leader while accelerating ResNet18 by 11.5% more than SFP [39] and FPGM [29]. For ResNet50, LRMF achieves the best performance on both TOP-1 accuracy and TOP-5 accuracy while maintaining a high acceleration rate. SFP [39] has the closest TOP-1 accuracy to LRMF but needs 10.9% more FLOPs. GDP [63] is able to accelerate ResNet50 to a similar level with LRMF, while TOP-1 accuracy loss is twice as bad as LRMF.

C. Parameter Analysis

Varied Pruned FLOPs. We conduct our LRMF on CIFAR-10 under five FLOPs reduction rates (*i.e.*, $\alpha = 0.1, 0.2, 0.3, 0.4, 0.5$) and the results are shown in Fig. 7. For all the networks, the overall trend of performance deteriorates as the reduction rate increases. Note that, the performance of ResNet110 pruned models surpass that of the unpruned baseline neural network, indicating the effectiveness of our LRMF on the over-parameterized neural network.

Pruning Interval. In our experiments, the pruning interval is set to 5 by default, *i.e.*, the pruning operation is performed at every 5 epochs of the training process. Here, we take ResNet56 accelerated on CIFAR-10 with 52.6% FLOPs reduction ($\alpha = 0.4$) as an example and vary the pruning interval within 10 in an odd form. As shown in Fig. 6 (a), the performance of the pruned network can approach the performance of the baseline network when pruning is carried out with 5 epochs as cycle. Note that this parameter can be adjusted by users to obtain the optimal results.

Number of Batch for DCT. Another important parameter is the amount of data used to find the geometric center of the learned representation in the frequency domain. Too much data will impose extra burdens on the training. By default, the training data of the two batches are used for this calculation, and shuffling is set True to guarantee fairness. We set up some comparative experiments for ResNet56 with a pruning rate of 0.4 to explain the effect of this parameter. As shown in Fig. 6 (b), the training data of the two batches are most suitable to conduct pruning. More data means more abundant data



Network	α	ΔFLOPs(%)$\downarrow$	ΔTop-1(%)$\downarrow$	Network	α	ΔFLOPs(%)$\downarrow$	ΔTop-1(%)$\downarrow$
	0.1	15.2	-0.10		0.1	14.9	0.26
	0.2	29.3	0.04		0.2	28.8	0.21
ResNet20	0.3	42.2	1.16	ResNet32	0.3	41.5	0.20
	0.4	54.0	1.73		0.4	53.2	0.55
	0.5	64.5	1.94		0.5	63.7	0.96
	0.1	14.7	0.12		0.1	14.6	-0.90
	0.2	28.4	-0.09		0.2	28.2	-0.54
ResNet56	0.3	41.1	0.79	ResNet110	0.3	40.8	-0.40
	0.4	52.6	0.30		0.4	52.3	-0.04
	0.5	63.2	0.99		0.5	62.8	0.39

Fig. 7. Influence of varied pruned FLOPs on pruning.

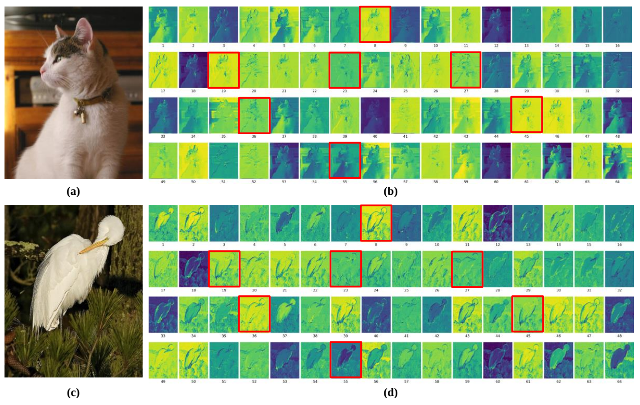


Fig. 8. Input images (left) and visualization of representations (right) of ResNet50-conv1. The filters corresponding to the representations with red bounding boxes are pruned when the pruning rate is 0.12.

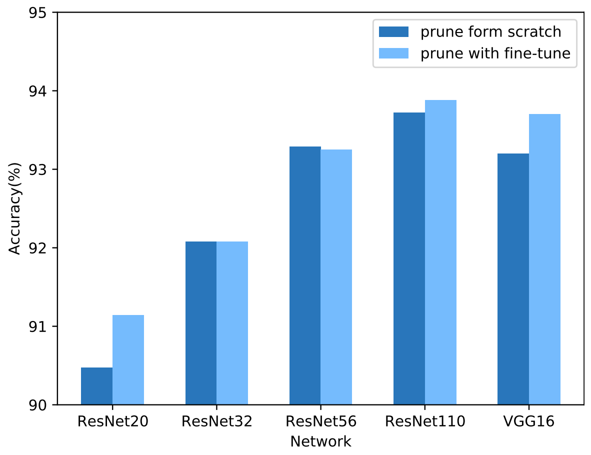


Fig. 9. Influence of fine-tuning on pruning.

representations, which needs more sufficient training. This is the reason that accuracy decreases as the number of batches increases.

D. Ablation Study

Further Explanation of Low-frequency Representations. Frequency information of an image is related to gradient in spatial domain. A region with slow changes in pixel values corresponds to low-frequency components, which is related to overall image information captured by human perception. Besides, when a neural network is optimized, more attention is paid to low-frequency components, which is related to generalization [65]. For a complete analysis, we apply our method on all representations or on high-frequency representations only. When the ResNet56 pruned with compress rate of 0.4 is taken as the baseline, this results in 93.29% accuracy on CIFAR-10 without fine-tuning. Not surprisingly, neither is as good as the baseline. Using all and high-frequency representations reduce the accuracy by 0.15% and 1.08%, respectively, which underlines the effectiveness of using low-frequency representations.

Fine-tuning. Our filter pruning method is experimented on two settings, *i.e.*, pruning from scratch and pruning with fine-tuning. As shown in Fig. 9, the influence of fine-tuning on pruning performance is uncertain, with less than 0.7% fluctuation in accuracy between the two different settings. Because of insensitivity to fine-tuning, users can choose different settings at their discretion.

Influence of Distance Type. In this paper, Euclidean distance, *i.e.*, ℓ_2-norm distance is utilized to calculate the similarity matrix $\mathbf{D}$ defined in Eq. (8). Here, we take the other two common distances into consideration, cosine and Manhattan (ℓ_1-norm) distance. We use the same baseline as before. The accuracy based on cosine and Manhattan distance is 92.86% and 92.92%, respectively, which indicates that these two distance metrics harm the performance of the pruned model to some extent.

Visualization of Learned Representations. The representations of the first convolution layer of the first residual block of ResNet50 trained on ImageNet are visualized in Fig. 8. When the pruning rate is set to 0.12, the filters with indexes (8, 19, 23, 27, 36, 45, 55) which correspond to the representations in red boxes are deactivated. We can easily find substitute pairs between these representations and the remaining ones, for example, (8, 19-17), (23-13), (27-20), (36-38), (45-46), and (55-54) in Fig. 8 (b), and (8-2, 1), (19-20), (23-25), (27-30), (36-39), (45-46), and (55-12) in Fig. 8 (d).

V. Conclusions

In this paper, we discuss the "relatively unimportant" criterion in the spatial domain and point out their limitations. To tackle these limitations, we propose a novel filter pruning method called LRMF, which for the first time determines absolutely unimportant filters by calculating the learned representation median in the frequency domain without fine-tuning. In particular, we mathematically define a representation function to ensure that the preserved filters maximize the characterization of the baseline neural networks. Then, we solve this representation function in the domain to further emphasize the perspective of "absolute unimportance", and the computational burden is also reduced because of the powerful energy compaction capability of the DCT. Thanks to these contributions, on various versions of CNNs, LRMF exhibits comparable performance to state-of-the-art pruning methods.

References

[1] Q. Xie, M. Luong, E. Hovy, and Q. V. Le, "Self-training with noisy student improves ImageNet classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. [Online]. Available: http://arxiv.org/abs/1911.04252

[2] J. Cao, H. Cholakkal, R. M. Anwer, F. S. Khan, Y. Pang, and L. Shao, "D2Det: Towards high quality object detection and instance segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 11 482–11 491.

[3] L. Porzi, M. Hofinger, I. Ruiz, J. Serrat, S. R. Bulo, and P. Kontschieder, "Learning multi-object tracking and segmentation from automatic annotations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. [Online]. Available: http://arxiv.org/abs/1912.02096

[4] M. Yousef and T. E. Bishop, "Origaminet: Weakly-supervised, segmentation-free, one-step, full page text recognition by learning to unfold," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. [Online]. Available: https://arxiv.org/abs/2006.07491

[5] Y. Tang, X. Yang, N. Wang, B. Song, and X. Gao, "Cgan-tm: A novel domain-to-domain transferring method for person re-identification," *IEEE Trans. Image Process.*, vol. 29, pp. 5641–5651, 2020.

[6] H. Guo, X. Yang, N. Wang, B. Song, and X. Gao, "A rotational libra r-cnn method for ship detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 8, pp. 5772–5781, 2020.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2016. [Online]. Available: https://arxiv.org/pdf/1512.03385

[8] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[9] A. Howard, M. Sandler, G. Chu, L. C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, and V. Vasudevan, "Searching for mobilenetv3," in *Proc. Int. Conf. Comput. Vis.*, 2019. [Online]. Available: http://arxiv.org/abs/1905.02244

[10] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "Ghostnet: More features from cheap operations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 1577–1586.

[11] Y. Idelbayev and M. A. Carreira-Perpiñán, "Low-rank compression of neural nets: Learning the rank of each layer," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 8046–8056.

[12] X. Zhang, J. Zou, X. Ming, K. He, and J. Sun, "Efficient and accurate approximations of nonlinear convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 1984–1992.

[13] D. Walawalkar, Z. Shen, and M. Savvides, "Online ensemble model compression using knowledge distillation," in *Proc. Eur. Conf. Comput. Vis.*, 2020. [Online]. Available: https://arxiv.org/abs/2011.07449

[14] T. Li, J. Li, Z. Liu, and C. Zhang, "Few sample knowledge distillation for efficient network compression," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. [Online]. Available: http://arxiv.org/abs/1812.01839

[15] Z. Liu, Z. Shen, M. Savvides, and K. Cheng, "ReActNet: Towards precise binary neural network with generalized activation functions," in *Proc. Eur. Conf. Comput. Vis.*, 2020. [Online]. Available: https://arxiv.org/abs/2003.03488

[16] A. Polino, R. Pascanu, and D. Alistarh, "Model compression via distillation and quantization," in *Proc. Int. Conf. Learn. Represent.*, 2018. [Online]. Available: http://arxiv.org/abs/1802.05668

[17] Z. Liu, H. Mu, X. Zhang, Z. Guo, X. Yang, K. Cheng, and J. Sun, "MetaPruning: Meta learning for automatic neural network channel pruning," in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 3295–3304.

[18] B. Li, B. Wu, J. Su, G. Wang, and L. Lin, "EagleEye: Fast sub-net evaluation for efficient neural network pruning," in *Proc. Eur. Conf. Comput. Vis.*, 2020. [Online]. Available: https://arxiv.org/abs/2007.02491

[19] X. Ding, Y. Guo, G. Ding, and J. Han, "Acnet: Strengthening the kernel skeletons for powerful cnn via asymmetric convolution blocks," in *ICCV*, 2019, pp. 1911–1920.

[20] D. W. Blalock, J. J. G. Ortiz, J. Frankle, and J. V. Guttag, "What is the state of neural network pruning?" *CoRR*, 2020. [Online]. Available: https://arxiv.org/abs/2003.03033

[21] F. Jonathan and C. Michael, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2019. [Online]. Available: http://arxiv.org/abs/1803.03635

[22] M. A. Carreira-Perpiñán and Y. Idelbayev, "Learning compression algorithms for neural net pruning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 8532–8541.

[23] T. Zhang, S. Ye, K. Zhang, J. Tang, W. Wen, M. Fardad, and Y. Wang, "A systematic dnn weight pruning framework using alternating direction method of multipliers," in *Proc. Eur. Conf. Comput. Vis.*, 2018. [Online]. Available: http://arxiv.org/abs/1804.03294

[24] L. Zeng and X. Tian, "Accelerating convolutional neural networks by removing interspatial and interkernel redundancies," *IEEE Trans. Cybern.*, vol. 50, no. 2, pp. 452–464, 2020.

[25] M. Lin, R. Ji, Y. Wang, Y. Zhang, B. Zhang, Y. Tian, and L. Shao, "HRank: Filter pruning using high-rank feature map," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. [Online]. Available: https://arxiv.org/abs/2002.10179

[26] S. Guo, Y. Wang, Q. Li, and J. Yan, "DMCP: Differentiable markov channel pruning for neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 1536–1544.

[27] Y. Zhou, G. G. Yen, and Z. Yi, "A knee-guided evolutionary algorithm for compressing deep neural networks," *IEEE Trans. Cybern.*, vol. 51, no. 3, pp. 1626–1638, 2021.

[28] X. Ding, G. Ding, Y. Guo, and J. Han, "Centripetal sgd for pruning very deep convolutional networks with complicated structure," in *CVPR*, 2019, pp. 4938–4948.

[29] Y. He, P. Liu, Z. Wang, Z. Hu, and Y. Yang, "Filter pruning via geometric median for deep convolutional neural networks acceleration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

[30] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient ConvNets," in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 4335–4344.

[31] K. Xu, M. Qin, F. Sun, Y. Wang, Y. Chen, and F. Ren, "Learning in the frequency domain," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. [Online]. Available: https://arxiv.org/abs/2002.12416

[32] J. Kim and S. Lee, "Deep learning of human visual sensitivity in image quality assessment framework," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 1969–1977.

[33] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," 2016. [Online]. Available: https://arxiv.org/abs/1602.07261

[34] N. Ma, X. Zhang, H. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient cnn architecture design," in *Proc. Eur. Conf. Comput. Vis.* [Online]. Available: https://arxiv.org/abs/1807.11164v1

[35] F. N. Iandola, M. W. Moskewicz, K. Ashraf, S. Han, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and ¡0.5mb model size," in *Proc. Int. Conf. Learn. Represent.*, 2017. [Online]. Available: http://arxiv.org/abs/1602.07360

[36] Y. Li, S. Gu, C. Mayer, L. V. Gool, and R. Timofte, "Group sparsity: The hinge between filter pruning and decomposition for network compression," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 8015–8024.

[37] S. Lin, R. Ji, C. Yan, B. Zhang, L. Cao, Q. Ye, F. Huang, and D. S. Doermann, "Towards optimal structured cnn pruning via generative adversarial learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. [Online]. Available: http://arxiv.org/abs/1903.09291

[38] J. Luo and J. Wu, "Neural network pruning with residual-connections and limited-data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 1455–1464.

[39] Y. He, G. Kang, X. Dong, Y. Fu, and Y. Yang, "Soft filter pruning for accelerating deep convolutional neural networks," in *IJCAI*, 2018. [Online]. Available: https://www.ijcai.org/proceedings/2018/0309

[40] J. Luo and J. Wu, "An entropy-based pruning method for CNN compression," 2017. [Online]. Available: http://arxiv.org/abs/1706.05791
[41] J. Ye, X. Lu, Z. Lin, and J. Z. Wang, "Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers," in *Proc. Int. Conf. Learn. Represent.*, 2018. [Online]. Available: http://arxiv.org/abs/1802.00124
[42] L. Gueguen, A. Sergeev, R. Liu, and J. Yosinski, "Faster neural networks straight from JPEG," in *Proc. Int. Conf. Learn. Represent.*, 2018. [Online]. Available: https://dl.acm.org/doi/abs/10.5555/3327144.3327308
[43] M. Ehrlich and L. Davis, "Deep residual learning in the JPEG transform domain," in *Proc. Int. Conf. Comput. Vis.*, 2019. [Online]. Available: http://arxiv.org/abs/1812.11690
[44] J. Yang, X. Fu, Y. Hu, Y. Huang, X. Ding, and J. Paisley, "PanNet: A deep network architecture for pan-sharpening," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 1753–1761.
[45] H. Wang, X. Wu, Z. Huang, and E. P. Xing, "High-frequency component helps explain the generalization of convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. [Online]. Available: http://arxiv.org/abs/1905.13545
[46] W. Xie, J. Lei, Y. Cui, Y. Li, and Q. Du, "Hyperspectral pansharpening with deep priors," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 5, pp. 1529–1543, 2020.
[47] Z. Liu, J. Xu, X. Peng, and R. Xiong, "Frequency-domain dynamic pruning for convolutional neural networks," in *Proc. Adv. Neural Inform. Process. Syst.*, 2018.
[48] Y. Wang, C. Xu, S. You, D. Tao, and C. Xu, "Cnnpack: Packing convolutional neural networks in the frequency domain," in *Proc. Adv. Neural Inform. Process. Syst.*, 2016.
[49] X. Dong, J. Huang, Y. Yang, and S. Yan, "More is less: A more complicated network with less inference complexity," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2017.
[50] X. Dong and Y. Yang, "Network pruning via transformable architecture search," in *Proc. Adv. Neural Inform. Process. Syst.*, 2019. [Online]. Available: http://arxiv.org/abs/1905.09717
[51] J. Zhong, G. Ding, Y. Guo, J. Han, and B. Wang, "Where to prune: Using lstm to guide end-to-end pruning," in *IJCAI*, 2018. [Online]. Available: https://doi.org/10.24963/ijcai.2018/445
[52] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 1895–1903.
[53] Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell, "Rethinking the value of network pruning," in *Proc. Int. Conf. Learn. Represent.*, 2019. [Online]. Available: http://arxiv.org/abs/1810.05270
[54] Y. He, X. Dong, G. Kang, Y. Fu, C. Yan, and Y. Yang, "Asymptotic soft filter pruning for deep convolutional neural networks," *IEEE Trans. Cybern.*, vol. 50, no. 8, pp. 3594–3604, 2020.
[55] Z. Zhuang, M. Tan, B. Zhuang, J. Liu, Y. Guo, Q. Wu, J. Huang, and J. Zhu, "Discrimination-aware channel pruning for deep neural networks," in *Proc. Adv. Neural Inform. Process. Syst.* [Online]. Available: https://arxiv.org/pdf/1810.11809
[56] S. Lin, R. Ji, C. Yan, B. Zhang, L. Cao, Q. Ye, F. Huang, and D. Doermann, "Towards optimal structured CNN pruning via generative adversarial learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 2785–2794.
[57] R. Yu, A. Li, C. Chen, J. Lai, V. I. Morariu, X. Han, M. Gao, C. Lin, and L. S. Davis, "NISP: Pruning networks using neuron importance score propagation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 9194–9203.
[58] Y. He, Y. Ding, P. Liu, L. Zhu, H. Zhang, and Y. Yang, "Learning filter pruning criteria for deep convolutional neural networks acceleration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 2006–2015.
[59] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *Handbook of Systemic Autoimmune Diseases*, 2009.
[60] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. FeiFei, "ImageNet large scale visual recognition challenge," *Proc. Int. J. Comput. Vis.*, 2015.
[61] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent.* [Online]. Available: https://arxiv.org/pdf/1409.1556
[62] P. Singh, V. K. Verma, P. Rai, and V. P. Namboodiri, "Leveraging filter correlations for deep model compression," in *WACV*, 2020, pp. 835–844.
[63] S. Lin, R. Ji, Y. Li, Y. Wu, and B. Zhang, "Accelerating convolutional networks via global & dynamic filter pruning," in *IJCAI*, 2018. [Online]. Available: http://www.ijcai.org/proceedings/2018/0336
[64] M. Kang and B. Han, "Operation-aware soft channel pruning using differentiable masks," in *ICML*, 2020. [Online]. Available: https://arxiv.org/abs/2007.03938
[65] W. Chen, J. T. Wilson, S. Tyree, K. Q. Weinberger, and Y. Chen, "Compressing convolutional neural networks," *CoRR*, 2015. [Online]. Available: http://arxiv.org/abs/1506.04449

Xin Zhang received the B.E. degree in Telecommunications Engineering from Xidian University, Xi'an, China in 2019. She is currently pursuing the Ph.D. degree with the Image Coding and Processing Center at State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an, China.

Her research interests include image processing, machine learning, and model compression.

Weiying Xie (M'18) received the B.S. degree in electronic information science and technology from university of Jinan in 2011. She received the M.S. degree in communication and information systems, Lanzhou University in 2014 and the Ph.D. degree in communication and information systems of Xidian University in 2017. Currently, she is an Associate Professor with the State Key Laboratory of Integrated Services Networks, Xidian University.

Her research interests include neural networks, machine learning, image processing, and high-performance computing.

Yunsong Li received the M.S. degree in telecommunication and information systems and the Ph.D. degree in signal and information processing from Xidian University, China, in 1999 and 2002, respectively. He joins the school of telecommunications Engineering, Xidian Uniersity in 1999 where he is currently a Professor. Prof. Li is the director of the image coding and processing center at the State Key Laboratory of Integrated Services Networks.

His research interests focus on image and video processing and high-performance computing.

Jie Lei received the M.S. degree in telecommunication and information systems and the Ph.D. degree in signal and information processing from Xidian University, China, in 2006 and 2010, respectively. He has been a Visiting Scholar at the Department of Computer Science of University of California, Los Angeles, USA, from 2014 to 2015. Currently, he is an Associate Professor at the school of Telecommunications Engineering, Xidian University, is a member of the image coding and processing center at the State Key Laboratory of Integrated Services Networks.

His research interests focus on image and video processing, computer vision, and customized computing for big-data applications.

Qian Du (S'98-M'00-SM'05-F'18) received the Ph.D. degree in electrical engineering from the University of Maryland, Baltimore, MD, USA, in 2000.

She is currently the Bobby Shackouls Professor in the Department of Electrical and Computer Engineering, Mississippi State University, Starkville, MS, USA. She is also an Adjunct Professor in the College of Surveying and Geo-informatics, Tongji University, Shanghai, China. Her research interests include hyperspectral remote sensing image analysis and applications, pattern classification, data compression, and neural networks.

Dr. Du is a Fellow of the IEEE-Institute of Electrical and Electronics Engineers, and a Fellow of the SPIE-International Society for Optics and Photonics. She received the 2010 Best Reviewer Award from the IEEE Geoscience and Remote Sensing Society. She was a Co-Chair of the Data Fusion Technical Committee of the IEEE Geoscience and Remote Sensing Society from 2009 to 2013, and the Chair of the Remote Sensing and Mapping Technical Committee of the International Association for Pattern Recognition from 2010 to 2014. She has served as an Associate Editor of the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING, the *Journal of Applied Remote Sensing*, and the IEEE SIGNAL PROCESSING LETTERS. Since 2016, she has been the Editor-in-Chief of the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING. She was the General Chair of the fourth IEEE GRSS Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing held at Shanghai, China, in 2012.