Efficient Harmonic Neural Networks With Compound Discrete Cosine Transform Filters and Shared Reconstruction Filters

Yao Lu, Le Zhang, Xiaofei Yang, and Yicong Zhou, *Senior Member, IEEE*

***Abstract*— The harmonic neural network (HNN) learns a combination of discrete cosine transform (DCT) filters to obtain an integrated feature from all spectra in the frequency domain. HNN, however, faces two challenges in learning and inference processes. First, the spectrum feature learned by HNN is insufficient and limited because the number of DCT filters is much smaller than that of feature maps. In addition, the number of parameters and the computation costs of HNN are significantly high because the intermediate spectrum layers are expanded multiple times. These two challenges will severely harm the performance and efficiency of HNN. To solve these problems, we first propose the compound DCT (C-DCT) filters integrating the nearest DCT filters to retrieve rich spectrum features to improve the performance. To significantly reduce the model size and computation complexity for improving the efficiency, the shared reconstruction filter is then proposed to share and dynamically drop the meta-filters in every frequency branch. Integrating the C-DCT filters with the shared reconstruction filters, the efficient harmonic network (EH-Net) is introduced. Extensive experiments on different datasets demonstrate that the proposed EH-Nets can effectively reduce the model size and computation complexity while maintaining the model performance. The code has been released at https://github.com/zhangle408/EH-Nets.**

***Index Terms*— Compound discrete cosine transform (C-DCT) filter, convolutional neural networks (CNNs), discrete cosine transform (DCT), harmonic neural networks (HNNs), shared reconstruction filter.**

NOMENCLATURE

DCT	Discrete cosine transform.
C-DCT	Compound discrete cosine transform.
$k \times k$	Spatial size of C-DCT filters.
κ	Compound level of C-DCT filters.
λ	Sliding step size of C-DCT filters.
l	Upper triangular selection.
l^*	Porous selection.
F	Number of C-DCT filters.
m	Number of meta-filters.
$\boldsymbol{\tau}$	Importance of meta-filters.
$\boldsymbol{\beta}$	Dropping rate of meta-filters.
$\boldsymbol{\eta}$	Indices of meta-filters.
$\boldsymbol{\rho}$	Binary masks of meta-filters.
$\boldsymbol{\psi}$	Selection indices for fusing filters.

Manuscript received 8 June 2021; revised 5 March 2022; accepted 15 May 2022. Date of publication 27 May 2022; date of current version 5 January 2024. This work was supported in part by the Guangdong Shenzhen Joint Youth Fund under Grant 2021A151511074, in part by the NSFC Fund under Grant 62176077, in part by the Shenzhen Key Technical Project under Grant 2020N046, and in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2019B1515120055. *(Corresponding author: Yicong Zhou.)*

Yao Lu is with the Department of Computer of Science and Technology, Harbin Institute of Technology (Shenzhen), Shenzhen 518055, China, and also with the Department of Computer and Information Science, University of Macau, Macau, China (e-mail: luyao2021@hit.edu.cn).

Le Zhang is with the Department of Computer of Science and Technology, Harbin Institute of Technology (Shenzhen), Shenzhen 518055, China (e-mail: zhangle408@gmail.com).

Xiaofei Yang and Yicong Zhou are with the Department of Computer and Information Science, University of Macau, Macau, China (e-mail: xiaofei.hitsz@gmail.com; yicongzhou@um.edu.mo).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2022.3176611.

Digital Object Identifier 10.1109/TNNLS.2022.3176611

I. INTRODUCTION

CONVOLUTIONAL neural networks (CNNs) [1], [2] have achieved tremendous successes in a wide variety of computer vision tasks, such as the image classification [3], detection of remote sensing [4], image retrieval [5], image denoising [6], and image captioning [7]. In the training process, CNNs learn filters to capture the local characteristics across channels in spatial dimension and can perform well when they can flexibly adjust to the data available. When the datasets contain limited training data, however, it will easily cause an overfitting problem in CNNs, leading to the poor generalization and performance degradation [8]. Different from CNNs, harmonic neural networks (HNNs) [8] were proposed to learn a combination of DCT filters to obtain an integrated feature from all spectra in the frequency domain. HNNs first utilize the preset DCT filters in every layer to retrieve spectrum features. Then, HNNs learn fusing filters with spatial size "1×1" to combine the spectrum information from all frequencies. Due to employing a collection of DCT filters with fixed parameters, HNNs have much fewer parameters compared to CNNs and thus can avoid the overfitting problem to some extent [8].

On the other hand, HNNs also bring disadvantages in terms of performance and efficiency.

1) *Performance Degradation:* HNNs replace every traditional convolutional layer in CNNs with the harmonic blocks. To keep the same number of parameters between CNNs and HNNs, in harmonic blocks, the spatial size of DCT filters is the same as to that of traditional convolutional filters. For example, the spatial size of convolutional filters in CNNs is usually set to "3×3."

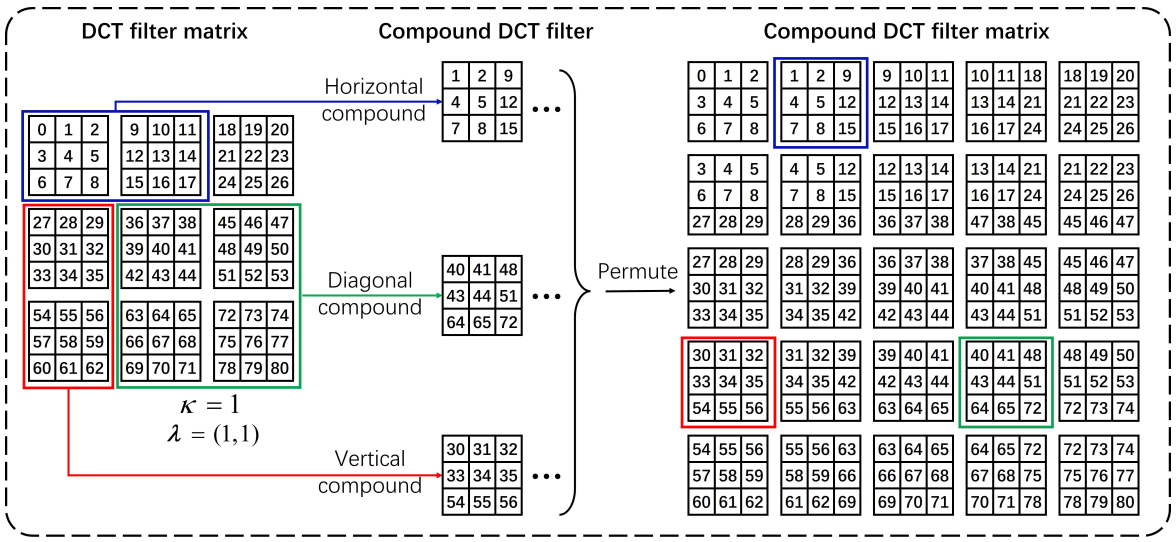

Fig. 1. Generation process of C-DCT filters based on DCT filters. The values of (compound) DCT filters are the orders of parameters. In this figure, the compound level κ of C-DCT filters is 1, and the sliding step $\boldsymbol{\lambda}$ for integrating DCT filters is (1, 1) in the horizontal and vertical directions. The boxes marked blue, red, and green represent the horizontal, vertical, and diagonal compound processes, respectively.

Hence, the spatial size "3 × 3" is also utilized in DCT filters. Compared to the spatial sizes of the feature maps, such as "224 × 224," the spatial size of DCT filters is much smaller. This implies that the spectrum features captured by DCT filters are not comprehensive enough because the same spatial sizes of DCT filters and feature maps can retrieve high-quality spectrum features according to DCT theory. In addition, since traditional HNNs can be compressed by decreasing the number of DCT filters, this will further lead to the performance degradation of HNNs.

2) *Efficiency Degradation:* Furthermore, according to [8], suppose that the spatial size of DCT filters is "$k \times k$," and the channels of the output spectrum feature maps are k^2 times larger than those of traditional convolutional layers. Thus, the subsequent fusing filters have many parameters even if their spatial size is only "1 × 1." Comparing traditional CNNs with similar performance, HNNs have significantly higher computation complexity from both the DCT and fusing layers, leading to the inefficiency of HNNs.

To address the first problem of performance degradation of HNNs, the key of solution is to improve the richness of DCT filters. However, traditional simply increasing the spatial size of DCT filters will largely promote the number of parameters in HNNs, leading to severe inefficiency. Due to this factor, this article proposes C-DCT filters to retrieve the rich spectrum features. The C-DCT filters shown in Fig. 1 integrate DCT filters with the nearest frequencies in the horizontal, vertical, and diagonal directions. Hence, the obtained spectrum feature will be more abundant and lead to better performance than traditional DCT filters.

For dealing with the network inefficiency, we will compress the fusing layers in HNNs. In recent years, many popular compression methods for traditional CNNs have been proposed. These compression methods can be broadly classified into the following categories.

1) Pruning methods [9]–[11] mainly drop the neurons, connections, or channels in the networks according to the specific principle of importance calculations. Several dynamic pruning methods [12], [13] were also proposed to dynamically prune the network connections or channels during the training process.
2) Quantizing methods [14]–[16] transfer the weights or features to low precision with fewer bits. This can decrease the memory space during calculations.
3) Group-convolution methods usually compress the networks using group convolutions to replace the traditional convolutions. Examples include the MobileNet family [17]–[19], interleaved grouped convolutional family [20]–[22], ShuffleNet family [23], [24] networks, and repeated group-convolution networks [25], [26]. Group convolutions divide the input channels and convolutional filters into the same number of groups. The filters in each group perform convolutions only on the input channels within the corresponding group. Thus, the parameters and calculations can be significantly decreased.
4) Network architecture searching methods [27]–[29] use reinforcement learning to automatically search the lightweight networks.
5) Knowledge distilling methods [30]–[32] use the teacher networks to supervise and train the student networks. The compressed networks can be obtained from the student networks after the training process.

Because of containing the most parameters of HNNs and dominating the computation complexity in the "1 × 1" fusing layer, such fusing layer will be compressed. However, traditional compression methods of pruning, quantizing, network architecture searching, and knowledge distilling usually suffers from complex training. The group convolutions are

widely applied to compress the "3×3" convolutional layers. In addition, the spectrum feature retrieved in different frequency branches will produce various importance to the final performance. These factors indicate that existing compression methods may be inappropriate to compress the fusing layers in HNNs. Furthermore, since the spatial size of such fusing layers is the smallest spatial size of "1×1," this will be difficult for compressing the fusing layers. Moreover, traditional simply decreasing the number of channels of fusing layers will also decrease the performance of HNNs.

Therefore, to address the second problem of inefficiency of HNNs, this article proposes the shared reconstruction filters to fuse the spectrum feature from all the frequencies. Such filters containing a small number of shared meta-filters significantly reduce the number of parameters of the fusing layer in HNNs. In order to reduce the computation complexity, we propose the dropping strategy to drop the shared meta-filters in some frequency branches that have less importance to performance. Applying the meta-filters to every frequency branch produces the shared spectrum feature. Finally, selecting and accumulating each shared spectrum feature map from all frequency branches will generate one output feature map. These dropping and shared computation processes can effectively reduce the computation cost of fusing layer in HNNs. Finally, integrating the proposed C-DCT filters with shared reconstruction filters, the efficient harmonic network (EH-Net) is further proposed in this article.

The contributions of this article are summarized as follows.

1) Integrating the DCT filters within the nearest frequencies, we propose the C-DCT filters to retrieve the rich spectrum feature and maintain the performance of compressed HNNs. The proposed C-DCT filters can also save the computation complexity to a satisfactory extent.
2) We propose the shared reconstruction filters that share a small number of meta-filters. To reduce the computation complexity, we further propose the dropping strategy and sharing calculations in the computation process. Such filters can effectively reduce the model size and computation complexity simultaneously.
3) Integrating the proposed C-DCT filters and shared reconstruction filters, we propose the EH-Net. Extensive experiment results demonstrate that the proposed EH-Nets can significantly reduce the model size and computation complexity while maintaining the performance of networks.

The rest of this article is organized as follows. Section II briefly reviews the theory of DCT and the structure of HNNs. Section III presents the proposed C-DCT filters, the shared reconstruction filters, and EH-Net. Section IV reports the experiment results on extensive datasets. Finally, Section V presents the conclusions of this work.

II. Related Works

A. Discrete Cosine Transform

Because of its property of compacting energy on natural images [33], DCT is widely employed to compress images and videos in JPEG and MPEG formats [34], respectively. The spatial frequency spectrum of an image can be produced through DCT decomposition. Performing the DCT to the image $\boldsymbol{x}$ with the width W and height H, the formulation is shown as follows:

$$\boldsymbol{y}_{u,v} = \sum_{i=0}^{H-1}\sum_{j=0}^{W-1}\sqrt{\frac{\alpha_u}{H}}\sqrt{\frac{\alpha_v}{W}}\boldsymbol{x}_{i,j} \times \cos\left(\frac{\pi}{H}\left(i+\frac{1}{2}\right)u\right)\cos\left(\frac{\pi}{W}\left(j+\frac{1}{2}\right)v\right) \quad (1)$$

where

$$\begin{cases} \alpha_u = 1,\ \alpha_v = 1, & \text{if } u = 0,\ v = 0 \\ \alpha_u = 2,\ \alpha_v = 2, & \text{otherwise.} \end{cases}$$

$\boldsymbol{y}_{u,v}$ denotes the DCT coefficient from the transformation of $\boldsymbol{x}$ with sinusoids at the uth and vth frequency in the horizontal and vertical directions, respectively. Equation (1) is similar to the convolutional operation if the term $\cos((\pi/H)(i + (1/2))u)\cos((\pi/W)(j + (1/2))v)$ is regarded as the (i, j)th parameter in the filter with size "$H \times W$." This term can be called the basis function as common practice. Inspired by this formulation, HNNs were proposed to employ this preset basis function to the traditional CNNs for generating the spectrum feature.

B. Harmonic Neural Networks

Currently, DCT has been applied to the deep learning models to improve the performance in many areas, especially for the image classification [35]–[37]. Different from the traditional methods, HNNs [8] were proposed to directly retrieve integrated features from all spectra in the frequency domain by learning the combinations of DCT filters. HNNs are constructed of harmonic blocks. Each harmonic block contains two layers, i.e., one DCT layer and one fusing layer. The DCT layer transforms the input feature into spatial spectrum information using the DCT basis function (DCT filters). This layer performs the DCT filters on the input feature map with a window-based strategy such as the convolutional operations. Suppose that the spatial kernel size is "$k \times k$," the number of DCT filters is k^2, and the DCT filters are denoted by $\boldsymbol{W}$ ($\boldsymbol{W} \in \mathbb{R}^{k^2 \times k \times k}$). Given the input feature $\boldsymbol{x}$ ($\boldsymbol{x} \in \mathbb{R}^{N \times H \times W}$) with channels N and spatial size "$H \times W$," then the output from the DCT operation with stride 1 is in the size "$k^2 \times N \times H \times W$." On the other hand, the second layer learns the weights for every spectrum feature channel using the combinational filter $\boldsymbol{\phi}$ with size "$M \times k^2 N \times 1 \times 1$," where M indicates the number of output channels. The calculation process of the ith output channel can be formulated as follows:

$$\begin{aligned} \boldsymbol{y}_i &= \boldsymbol{\phi}_i \otimes \left(\mathbf{C}_{f=0}^{k^2-1}\mathbf{C}_{s=0}^{N-1}\boldsymbol{W}_f \otimes \boldsymbol{x}_s\right) \\ &= \sum_{s=0}^{N-1}\sum_{f=0}^{k^2-1}\boldsymbol{\phi}_{i,s,f}^{r}\boldsymbol{W}_f \otimes \boldsymbol{x}_s = \sum_{s=0}^{N-1}\sum_{f=0}^{k^2-1}\left(\boldsymbol{\phi}_{i,s,f}^{r}\boldsymbol{W}_f\right) \otimes \boldsymbol{x}_s \end{aligned} \quad (2)$$

where $\otimes$ indicates the 2-D convolutional operation, $\mathbf{C}$ denotes the concatenation operation at the channel axis, and $\boldsymbol{\phi}^r$ represents the reshaped combinational filter with size "$M \times N \times k^2$." In the HNNs, the spatial size of DCT filters is the same

to that of traditional convolutional filters, such as "3×3." Furthermore, the model size of harmonic networks reduces when dropping some DCT filters with high frequencies. Due to only learning the collection of DCT filters with fixed parameters, HNNs have much fewer parameters compared to CNNs and thus can avoid the overfitting problem to some extent [8]. Therefore, our work will focus on simultaneously improving the performance and efficiency of HNNs.

III. Compound DCT Filters

In order to capture much more abundant spectrum information from input features, this section proposes the C-DCT filters. To generate the C-DCT filters, we combine two adjacent basic DCT filters in the horizontal and vertical directions while integrating four adjacent DCT filters in the diagonal direction. An illustrative example of the proposed C-DCT filters is shown in Fig. 1.

Suppose that the spatial width or height of basic DCT filters is k, and the number of basic plane DCT filters is "$k \times k$." Therefore, from the definition of DCT [see (1)], the formulation of basic DCT filters $\boldsymbol{W}^b$ is shown as follows:

$$\boldsymbol{W}^b = \begin{Bmatrix} \boldsymbol{W}^b_{0,0} & \boldsymbol{W}^b_{0,1} & \cdots & \boldsymbol{W}^b_{0,k-1} \\ \boldsymbol{W}^b_{1,0} & \boldsymbol{W}^b_{1,1} & \cdots & \boldsymbol{W}^b_{1,k-1} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{W}^b_{k-1,0} & \boldsymbol{W}^b_{k-1,1} & \cdots & \boldsymbol{W}^b_{k-1,k-1} \end{Bmatrix} \tag{3}$$

where $\boldsymbol{W}^b \in \mathbb{R}^{k\times k\times k\times k}$ and $\boldsymbol{W}^b_{i,j} \in \mathbb{R}^{k\times k}$ ($i, j \in \{0, 1, \ldots, k-1\}$).

To demonstrate the integrating process easily, the C-DCT filters can be generated by the following steps. First, the basic DCT filters will be transposed in the second and third dimensions. Then, the transposed DCT filters will be reshaped from size "$k \times k \times k \times k$" to size "$k^2 \times k^2$," obtaining DCT filter matrix $\boldsymbol{W}^{b*}$ ($\boldsymbol{W}^{b*} \in \mathbb{R}^{k^2\times k^2}$). Finally, the C-DCT filters can be produced by sliding at the reshaped DCT filters with a sliding window size "$k \times k$" and then cropping the values from the sliding windows.

Based on Fig. 1, the number of the generated C-DCT filters between two basic DCT filters is called the compound level, denoting as κ ($\kappa \in \{0, 1, 2, \ldots, k-1\}$). Thus, the generated C-DCT filters are "$(k-1) \times (\kappa+1)+1$." In particular, the C-DCT filter matrix degrades to the DCT filter matrix when $\kappa = 0$. Furthermore, the C-DCT filters are also generated according to the sliding step size in the group of DCT filters to be integrated. $\boldsymbol{\lambda}$ represents this sliding step size, where $\boldsymbol{\lambda} = \{\boldsymbol{\lambda}_0, \boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_\kappa\}$ and $\boldsymbol{\lambda}_i = (\lambda^h_i, \lambda^w_i)$ ($i \in \{0, 1, 2, \ldots, \kappa\}$, $\lambda^h_i, \lambda^w_i \in \{0, 1, 2, \ldots, k-1\}$). According to the above definition, $\boldsymbol{\lambda}_i$ indicates the sliding step size at the vertical and horizontal directions. In particular, $\boldsymbol{\lambda}_0 = (0, 0)$ by default indicates that the C-DCT filter matrix reverts to the original DCT filter matrix. In summary, the number of C-DCT filters is determined by the compound level κ, while the values of C-DCT filters are generated through the sliding step size $\boldsymbol{\lambda}$ relative to the basic DCT filters.

From the above demonstrations, the final C-DCT filter matrix $\boldsymbol{W}^c$ can be formulated by the following equation:

$$\boldsymbol{W}^c = \begin{Bmatrix} \boldsymbol{W}^c_{0,0} & \boldsymbol{W}^c_{0,1} & \cdots & \boldsymbol{W}^c_{0,k^*-1} \\ \boldsymbol{W}^c_{1,0} & \boldsymbol{W}^c_{1,1} & \cdots & \boldsymbol{W}^c_{1,k^*-1} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{W}^c_{k^*-1,0} & \boldsymbol{W}^c_{k^*-1,1} & \cdots & \boldsymbol{W}^c_{k^*-1,k^*-1} \end{Bmatrix} \tag{4}$$

where $\boldsymbol{W}^c \in \mathbb{R}^{k^*\times k^*\times k\times k}$ ($k^* = (k-1)\times(\kappa+1)+1$) and $\boldsymbol{W}^c_{i,j} \in \mathbb{R}^{k\times k}$ ($i, j \in \{0, 1, 2, \ldots, k^*-1\}$). Specifically, in (4), $\boldsymbol{W}^c_{i,j}$ is shown as follows:

$$\begin{aligned} \boldsymbol{W}^c_{i,j} &= \boldsymbol{W}^{b*}_{i^*:i^*+k,\ j^*:j^*+k} \\ i^* &= \left\lfloor \frac{i}{\kappa+1} \right\rfloor \times k + \lambda^h_{(i \bmod (\kappa+1))} \\ j^* &= \left\lfloor \frac{j}{\kappa+1} \right\rfloor \times k + \lambda^w_{(j \bmod (\kappa+1))} \end{aligned} \tag{5}$$

where "$i^* : i^*+k$ and $j^* : j^*+k$" indicate the top-left and bottom-right coordinates, respectively, and the sliding windows on $\boldsymbol{W}^{b*}$ are (i^*, j^*) and (i^*+k-1, j^*+k-1). In particular, in (5), when $i \bmod (\kappa+1) = 0$ and $j \bmod (\kappa+1) = 0$, $\boldsymbol{W}^c_{i,j} = \boldsymbol{W}^b_{\lfloor i/(\kappa+1)\rfloor, \lfloor j/(\kappa+1)\rfloor}$. This implies that, except these basic DCT filters, other filters in $\boldsymbol{W}^c$ are all constructed by cropping values in sliding windows. Fig. 1 shows the process of producing C-DCT filters with compound level $\kappa = 1$ and sliding step size $\boldsymbol{\lambda} = (1, 1)$. After cropping filters based on (5), all the basic DCT filters and C-DCT filters are permuted to produce the C-DCT filter matrix. From Fig. 1, this compound process can generate much more abundant DCT filters with compound frequencies. This can integrate the basic information with other features from compound frequency spectrums, improving richness of the captured feature.

Finally, two efficient strategies are introduced to reduce computation complexity.

1) *Upper Triangular Selection:* The information captured by low-frequency DCT filters usually produces a heavier influence on the final performance of networks than the information captured by high-frequency DCT filters. Therefore, similar to the selection manner in [8], on the easy datasets, we can select the C-DCT filters from the upper triangular areas of C-DCT filter matrix. The number of upper triangular C-DCT filters is determined by the selection level l, where $l \in \{1, 2, \ldots, k^*+1\}$. The selection process is shown in Fig. 2(a). According to (4), the C-DCT filter $\boldsymbol{W}^u$ ($\boldsymbol{W}^u \in \mathbb{R}^{k^*\times k^*\times k\times k}$) with upper triangular selection is formulated as follows:

$$\boldsymbol{W}^u_{i,j} = \begin{cases} \boldsymbol{W}^c_{i,j}, & i+j < l \\ \boldsymbol{0}, & \text{otherwise.} \end{cases} \tag{6}$$

2) *Porous Selection:* On the difficult datasets, it will decrease the performance of networks when discarding the information retrieved from high-frequency DCT filters. We then propose another selection strategy called porous selection, as shown in Fig. 2(b). In this figure, the filters marked in blue are to be selected. The number of C-DCT filters from porous selection

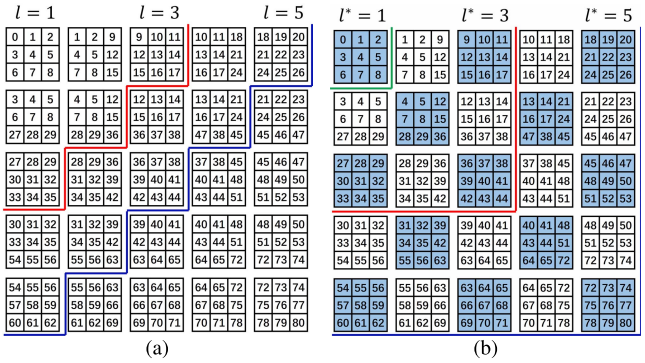

Fig. 2. Comparison between upper triangular selection and porous selection. In (a), l indicates the selected level for upper triangular. In (b), l^* denotes the selected level for porous selection. The selected filters at the corresponding level in (b) are marked by blue. (a) Upper triangular selection. (b) Porous selection.

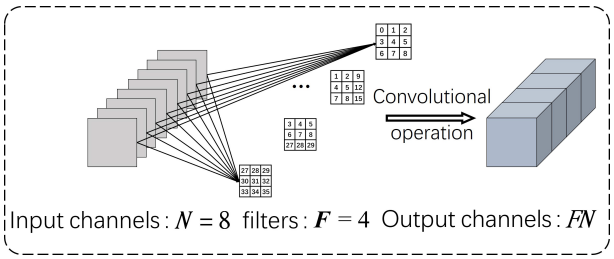

Fig. 3. Harmonic convolution between the C-DCT filters and input channels.

is determined by the selected level l^*, where $l^* \in \{1, 2, \ldots, k^*+1\}$. According to (4), the C-DCT filter $\boldsymbol{W}^p$ ($\boldsymbol{W}^p \in \mathbb{R}^{k^* \times k^* \times k \times k}$) with porous selection is formulated as follows:

$$\boldsymbol{W}^p_{i,j} = \begin{cases} \boldsymbol{W}^c_{i,j}, & (i+j) \bmod 2 = 0) \wedge \left(i,\, j < l^*\right) \\ \boldsymbol{0}, & \text{otherwise.} \end{cases} \tag{7}$$

This demonstrates a porous selection method can utilize much more C-DCT filters at high frequencies. Utilizing these two selection methods, the networks can achieve satisfactory performance on datasets with different difficulties. Furthermore, it can also effectively reduce the computation complexity for networks.

Selecting the compound filters at both low and high frequencies, we obtain the final C-DCT filters. Then, the C-DCT filters can be easily applied to the input channels by traditional convolutional operations. The convolutional operation is conducted between each C-DCT filter and each input channel. Suppose that the numbers of input channels and C-DCT filters are represented by N and F, respectively, and the number of output channels is "$F \times N$." The process of the convolutional operations between the input channels and C-DCT filters is called harmonic convolutions, as shown in Fig. 3.

IV. Shared Reconstruction Filters

This section first introduces the structure of the shared reconstruction filters and then proposes two dropping strategies to improve the computation complexity of the shared reconstruction filters.

A. Filter Structure

In the next stage, the networks will utilize learnable filters to fuse the information retrieved in different frequencies. Since the spatial size of fusing layers is the smallest spatial size of "1×1," this leads to difficulty in compressing the fusing layers. Furthermore, traditional simply reducing the number of channels of fusing layers will result in poor fused features from spectrum features and also decrease the performance of HNNs. For reducing the parameters and computation complexity in this fusing stage while preserving the channel richness, inspired by repeated group convolution, a novel filter called shared reconstruction filters is proposed. Specifically, this shared reconstruction filter is constructed through sharing meta-filters along the channel axis. The number of meta-filters is much smaller than that of the traditional fusing filters. This significantly reduces the learnable parameters while preserving the channel richness. Moreover, the calculations of this fusing convolutional operation are also shared based on the shared meta-filters. To further reduce the computation complexity, the meta-filters in some specific frequency branches can be dropped through the proposed dropping strategy. This saves the calculations of shared meta-feature generated by shared meta-filters and thus reduces the overall computation complexity.

Given the input feature with channels N and C-DCT filters with F frequencies, according to Fig. 3, FN output channels will be produced by the harmonic convolution. Suppose that the number of output channels after fusing features from different frequencies is M, and then, the number of parameters in traditional fusing filters is $FN \times M$. However, in the proposed shared reconstruction filters, only m meta-filters are shared and reconstructed, where $m = \alpha M$ ($\alpha \in (0, 1]$). This indicates that $m \leq M$ and the number of parameters in the proposed filters is $FN \times m$. The smaller α is, the more parameters are reduced in the proposed filters. The process of reconstructing filters from shared meta-filters is shown in Fig. 4.

As shown in Fig. 4, FN feature channels are first divided into F groups. Then, m meta-filters $\boldsymbol{\omega}_s$ ($\boldsymbol{\omega}_s \in \mathbb{R}^{m \times N \times 1 \times 1}$) are selected to reconstruct M fusing filters $\boldsymbol{\omega}_r$ ($\boldsymbol{\omega}_r \in \mathbb{R}^{M \times N \times 1 \times 1}$). The meta-filters are shared and applied to each frequency group. Since the feature obtained by some frequency DCT filters is not important for the final performance of networks, the number of shared meta-filters can be reduced in those frequency branches. For example, the feature retrieved from high-frequency DCT filters has a negligible impact to the feature quality. Thus, such high-frequency feature dose not need abundant meta-filters. This implies that some unimportant meta-filters can be dropped in these high-frequency feature branches. Through this manner of dropping meta-filters, the computation complexity can be further reduced in the calculations of the shared feature. At the dropping process, the dropping rate $\boldsymbol{\beta}$ ($\boldsymbol{\beta} = \{\beta_0, \beta_1, \ldots, \beta_{F-1}\},\ \beta \in [0, 1]$) controls the removal of meta-filters. Inspired by [38], L_1 norm of weights of each meta-filter can be applied to compute the importance of meta-filters. Therefore, for the meta-filters $\boldsymbol{\omega}_s$, their importance is denoted by $\boldsymbol{\tau} = \{\tau_0, \tau_1, \ldots, \tau_{m-1}\}$, where

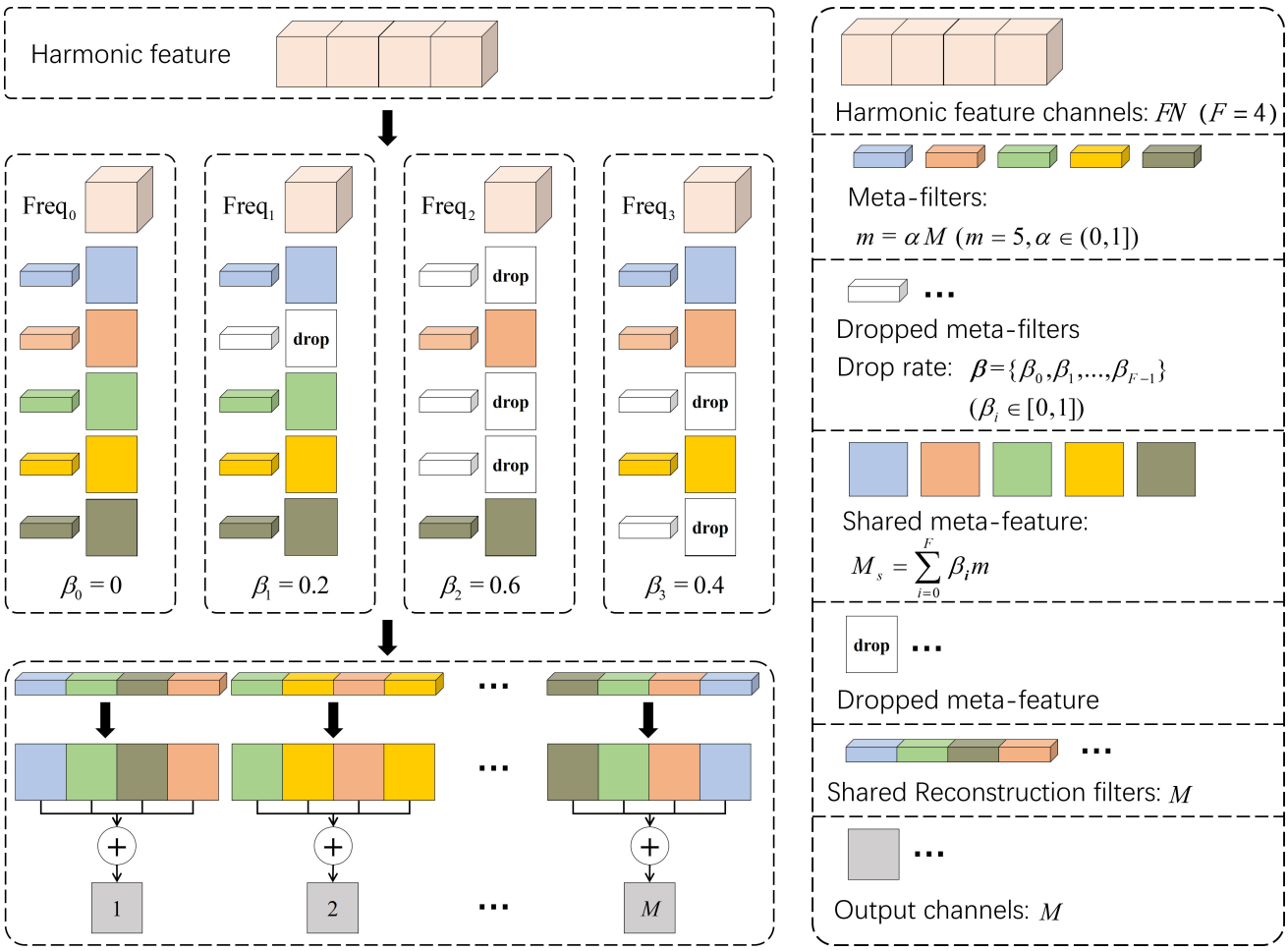

Fig. 4. Generation of the output feature using the shared reconstruction filters. N and M indicate the input and output channels, respectively. F denotes the number of the C-DCT filters (frequency branches) in harmonic convolution. Based on Fig. 3, the harmonic convolution produces FN channels of the spectrum feature. In this figure, the spectrum input feature is divided into F groups according to frequency. Then, m ($m = \alpha M, \alpha \in (0, 1]$) shared meta-filters are applied to every frequency group, generating shared meta-feature from all groups. In this step, the meta-filters can be further dropped in some specific groups with a drop rate $\boldsymbol{\beta}$ ($\boldsymbol{\beta} = \{\beta_0, \beta_1, \ldots, \beta_{F-1}\}, \beta_i \in [0, 1], i \in 0, 1, .., F-1$) to reduce the computation complexity, such as the white boxes in the shared meta-feature. Finally, M shared reconstruction filters are reconstructed by selecting the meta-filters group by group and then concatenating these selected meta-filters along the axis of input channels. In the computation process, M output feature channels are produced through selecting and summating the shared meta-feature group by group according to the selection index of the shared reconstructed filters. In this figure, the meta-filters with the same color share the same parameters.

τ_i ($i \in \{0, 1, \ldots, m-1\}$) is formulated as follows:

$$\tau_i = \left\|\boldsymbol{\omega}_s^i\right\|_1 = \sum_{k=0}^{N-1} \left|\omega_s^{i,k}\right|. \tag{8}$$

According to the importance $\boldsymbol{\tau}$ and dropping rate $\boldsymbol{\beta}$ of meta-filters, the dropped meta-filters in every frequency branch can be determined. First, the importance of meta-filters $\boldsymbol{\tau}$ is sorted with an increasing order. We obtain the indices of the meta-filters with an increasing order of importance, denoted by $\boldsymbol{\eta}$ ($\boldsymbol{\eta} = \{\eta_0, \eta_1, \ldots, \eta_{m-1}\}$), where η_i ($i \in \{0, 1, \ldots, m-1\}$) is the index of the ith least importance meta-filter. Next, a binary mask $\boldsymbol{\rho}$ ($\boldsymbol{\rho} \in \mathbb{R}^{F \times m}$, $\boldsymbol{\rho} = \{\boldsymbol{\rho}_0, \boldsymbol{\rho}_1, \ldots, \boldsymbol{\rho}_{F-1}\}^T$) can be utilized to remove the unimportant meta-filters in every frequency branch, where $\boldsymbol{\rho}_i$ can be formulated as follows:

$$\boldsymbol{\rho}_i = \left\{\rho_i^0, \rho_i^1, \ldots, \rho_i^{m-1}\right\}, \quad \rho_i^j = \begin{cases} 0, & j \in \boldsymbol{\eta}_{[0:\xi_i]} \\ 1, & \text{otherwise} \end{cases} \tag{9}$$

where $\xi_i = \lfloor \beta_i m \rfloor$ and $\boldsymbol{\eta}_{[0:\xi_i]}$ represents the first ξ_i values in $\boldsymbol{\eta}$.

After dropping the meta-filters in every frequency branch, the remaining meta-filters can be reconstructed. For every reconstructed filter, a binary mask tensor is utilized to select only one meta-filter from every frequency branch. Concatenating these selected meta-filters forms the final shared reconstruction filter. Therefore, for all the reconstructed filters $\boldsymbol{\omega}^r$ with size "$M \times N \times 1 \times 1$," the selection binary mask tensor can be denoted by $\boldsymbol{\psi}$ with size "$M \times F \times m$," where $\|\boldsymbol{\psi}_{\cdot,i}\|_0 = 1$. This implies that the selection binary mask vector in every frequency branch has only one nonzero value of 1 and all other values are 0s. Finally, the reconstructed filters can be formulated as follows:

$$\boldsymbol{\omega}_i^r = (\boldsymbol{\psi}_i * \boldsymbol{\rho}) \otimes \boldsymbol{\omega}^s \tag{10}$$

where $\boldsymbol{\omega}_i^r$ ($\boldsymbol{\omega}_i^r \in \mathbb{R}^{1 \times N \times 1 \times 1}, i \in \{0, 1, \ldots, M-1\}$) is the ith reconstructed filter in $\boldsymbol{\omega}^r$ and $*$ and $\otimes$ indicate Hadamard and outer products, respectively.

B. Improvement of Computation Efficiency

Since the shared reconstruction filters are generated from meta-filters, some shared reconstruction filters may select the same meta-filters at the same positions. This results in duplicated calculations in the computation process. For example, in Fig. 4, the second meta-filters in the first and last shared reconstruction filters are the same. Therefore, when applying the convolutional operations to the input and its meta-filter, the calculations will be repeated multiple times.

In order to reduce the duplicate calculations, we can share the computations between each meta-filter and its corresponding input branch. Specifically, the computation process between the shared reconstruction filters and input channels can be formulated as follows:

$$\begin{aligned} \boldsymbol{y}_i &= \sum_{j=0}^{F-1} \boldsymbol{x}_j \otimes \left(\left(\boldsymbol{\psi}_j * \boldsymbol{\rho} \right) \otimes \boldsymbol{\omega}^s \right) \\ &= \sum_{j=0}^{F-1} \left(\boldsymbol{\psi}_j * \boldsymbol{\rho} \right) \otimes \left(\boldsymbol{x}_j \otimes \boldsymbol{\omega}^s \right) \end{aligned} \tag{11}$$

where $\boldsymbol{y}_i$ denotes the ith output channel and $\boldsymbol{x}_j$ is the jth input frequency branch. From (11), in the computation process, we can compute all convolutions between the meta-filters and input branches. All output features can be called meta-features. Then, the same selection indices $\boldsymbol{\psi}$ of shared reconstruction filters can be used to select the corresponding shared meta-feature. Through this manner, the duplicate computations can be avoided. To further reduce the computation complexity, the dropping strategy can be used to reduce the unnecessary meta-filters from every branch. Thus, the calculations between these unimportant meta-filters and their corresponding input frequency branches can be removed to further improve the efficiency. Then, (11) can be transformed as follows:

$$\begin{aligned} \boldsymbol{y}_i &= \sum_{j=0}^{F-1} \boldsymbol{x}_j \otimes \left(\left(\boldsymbol{\psi}_j * \boldsymbol{\rho} \right) \otimes \boldsymbol{\omega}^s \right) \\ &= \sum_{j=0}^{F-1} \boldsymbol{\psi}_j \otimes \left(\boldsymbol{x}_j \otimes \left(\boldsymbol{\rho} * \boldsymbol{\omega}^s \right) \right). \end{aligned} \tag{12}$$

Dropping meta-filters in each frequency branch and sharing meta-feature in every output channel, the computation complexity can be significantly reduced.

V. Efficient Harmonic Networks

After introducing the basic efficient harmonic block, this section proposes the EH-Nets and then analyzes the parameters and computation of floating-point operations (FLOPs) of EH-Nets.

A. Basic Efficient Harmonic Block

The proposed EH-Nets consist of multiple basic efficient harmonic blocks. Each block is constructed according to the traditional convolutional layer. Suppose that the traditional convolutional layer contains the convolutional filters with size "$k \times k$," and then, in each basic efficient harmonic block, the traditional convolutional layer will be replaced with the C-DCT filters with size "$k \times k$" and one fusing layer with the shared reconstruction filters. A batch normalization layer is inserted between the C-DCT filter layer and the shared reconstruction filter layer. Fig. 5 shows the structure of a basic efficient harmonic block. Furthermore, the detailed calculation process of the basic efficient harmonic block is shown in Algorithm 1.

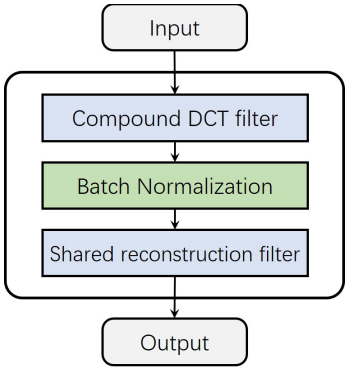

Fig. 5. Structure of a basic efficient harmonic block.

B. Efficient Harmonic Networks

The proposed EH-Nets of EH-Nets are implemented using traditional networks, such as VGG-Nets [39] and ResNets family networks [40], [41]. In order to keep the consistence of the traditional networks, the basic framework of EH-Nets keeps the same as that of the regular popular networks. Furthermore, the spatial kernel size in every C-DCT filter layer is also set to the same as the traditional convolutional kernels. For example, the kernel size is usually set to "3×3." The number of output channels in basic efficient harmonic blocks should be the same as that in the traditional regular convolutional layers. Based on these principles, the proposed EH-Nets can be easily constructed by simply replacing every regular convolutional layer with basic efficient harmonic block, as demonstrated in Algorithm 1.

C. Parameters and FLOPs of EH-Nets

1) Parameter Analysis: Since EH-Nets simply replace the traditional convolutional layer using the basic efficient harmonic blocks, the number of parameters and FLOPs of EH-Nets can be computed on one layer. Suppose that the number of input and output channels is N and M, respectively. The spatial kernel size is "$k \times k$." Therefore, the number of parameters P_r in the traditional convolutional layer is

$$P_r = M \times N \times k \times k = MNk^2. \tag{13}$$

The traditional harmonic network is composed of one DCT layer and one "1×1" fusing layer. Suppose that the number of DCT filters is F, and the number of output channels is FN. Thus, the number of parameters in a harmonic block P_h is

$$P_h = M \times FN \times 1 \times 1 = MFN. \tag{14}$$

The proposed efficient harmonic block contains one C-DCT filter layer and one shared reconstruction filter layer.

Algorithm 1 Algorithm of Basic Efficient Harmonic Block

1: Initialize: Spatial size of C-DCT filters $k \times k$;
Compound level κ and sliding step size λ of C-DCT filters;
Selection methods (upper triangular or porous selection) of C-DCT filters;
Selection level (l or l^*) of C-DCT filters;
Number of C-DCT filters $F = f(l)$ or $F = f(l^*)$;
Number of out feature channel M, Number of meta-filters $m = \alpha M$;
Parameters of meta-filters $\boldsymbol{\theta}$;
Dropping rate of meta-filters $\boldsymbol{\beta}$;
Learnable selection indices for reconstructed fusing filters $\boldsymbol{\psi}$.
2: **Repeat:**
3: Harmonic convolutional layer input: $\boldsymbol{x}$;
4: Generate C-DCT filters based on Eqns. (3) to (5);
5: Select F C-DCT filters according to initialized selection method and selection level based on Eqns. (6) and (7);
6: Determine the dropped meta-filters in every frequency branch based on Eqns. (8) and (9);
7: Compute the shared meta-feature in every frequency branch based on Eqn. (11);
8: Select meta-feature from every frequency branch to formulate the final output feature based on Eqn. (12);
9: Compute loss and update $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$.
10: **Until convergence**

The parameters are dependent on the meta-filters $\boldsymbol{w}_s$ and the selection indices $\boldsymbol{\psi}$ for the shared reconstruction filters. Since the selection indices are binary values, they can be saved by bits in the inference process. This implies that the required memory of $\boldsymbol{\psi}$ is very small and thus can be ignored compared with the other parameters. Suppose that the number of C-DCT filters is F' in the basic efficient harmonic block, and according to (10), the number of parameters P_e is

$$\begin{aligned} P_e &= \alpha M \times N \times 1 \times 1 + M \times F \times \alpha M \text{ (bits)} \\ &\approx \alpha MN. \end{aligned} \tag{15}$$

In particular, based on (15), different from the traditional harmonic networks, the number of parameters of EH-Nets has no relationship with the number of C-DCT filters. Thus, the model size of EH-Nets will not be influenced by the number of C-DCT filters. Based on (13)–(15), the following equation can be obtained:

$$P_e = \frac{\alpha}{k^2} P_r = \frac{\alpha}{F} P_h \tag{16}$$

where $\alpha \in (0, 1]$. In Section VI, $\alpha = 0.5$, $k = 3$, and $F = 9$, and then, $P_e = (1/18)P_r = (1/18)P_h$. This demonstrates that the memory cost can be significantly reduced in theory. This further indicates that the EH-Nets can retrieve much more abundant frequency spectrum features while reducing the model size.

2) FLOPs Analysis: According to the settings of the input channels, output channels, and the kernel size in Section V-C1, the computation FLOPs F_r of traditional convolutional layer is

$$F_r = M \times N \times k \times k \times w \times h = k^2 whMN. \tag{17}$$

The harmonic computations are composed of the kernel construction and convolutional operations [8]. The number of FLOPs P_h can be formulated as follows:

$$\begin{aligned} P_h &= M \times N \times F \times k \times k \\ &\quad + M \times N \times k \times k \times w \times h \\ &= k^2 MN(F + wh). \end{aligned} \tag{18}$$

The proposed efficient harmonic block includes the computations of the C-DCT filters, shared meta-filters, and feature fusion. Furthermore, the computations of fusing meta-feature are also included in the calculation process. As the implementations in Section Vi, the drop rate linearly increases from 0 to β^* in F' frequency branches. Therefore, the number of FLOPs of the basic efficient harmonic block F_e is shown as follows:

$$\begin{aligned} F_e &= w \times h \times \Big(N \times F' \times k^2 \\ &\quad + \sum_{i=0}^{F'-1} \lceil (1 - \beta_i)\alpha M \rceil \times N + F' \times M \Big) \\ &= whF' \left(k^2 N + M + \frac{\alpha(2 - \beta^*)}{2} MN \right) \end{aligned} \tag{19}$$

where F' is the number of C-DCT filters. Based on (19), we can obtain

$$\begin{aligned} F_e &= F' \left(\frac{1}{M} + \frac{1}{k^2 N} + \frac{\alpha(2 - \beta^*)}{2k^2} \right) F_r \\ &\approx \frac{\alpha(2 - \beta^*)F'}{2k^2} F_r. \end{aligned} \tag{20}$$

If $F' = 15$, $\beta^* = 0.4$, $\alpha = 0.5$, and $k = 3$, then, $F_e \approx (2/3)F_r$. This indicates that the proposed efficient harmonic block effectively reduces the computation FLOPs.

In summary, the proposed EH-Nets can significantly decrease the model size and calculation of FLOPs.

VI. Experiments and Analysis

In this section, datasets and initialization will be first introduced in Section VI-A. The different components in the proposed EH-Nets will be compared and verified the effectiveness of these components in Sections VI-B–VI-D. Section VI-E compares the proposed EH-Nets with the baseline harmonic

networks on various datasets from the aspects of model size, computation FLOPs, and accuracy.

A. Datasets and Initializations

The proposed EH-Nets are compared with traditional networks on the following datasets.

1) Regular ImageNet Datasets: The regular ILSVRC 2012 classification dataset [42] is composed of more than 1.2 million training images and 50 000 validation images in 1000 categories. The data augmentations in [22], [43], and [44] are applied and a single crop with the size "224×224" is utilized at the inference. The classification accuracies are reported on the validation set.

2) CIFAR Datasets: CIFAR datasets contain all images in CIFAR-10 [45] and CIFAR-100 [46] with 10 and 100 classes, respectively. They both have 60 000 colored nature scene images with the resolution "32×32." There are 50 000 training images and 10 000 test images. Data augmentation is the same as in [40], [43], and [47].

3) CINIC-10 Datasets: CINIC-10 dataset [48] is an extension of CIFAR-10 through the supplement from downsampled ImageNet images. It contains 270 000 colored images with resolution "32×32" in ten categories. The training, validation, and test subsets are equal-sized. As the illustrations in [48], the regular ImageNet is an unwieldy dataset because the spatial size of images is large and a single training run is very time-consuming without sufficient computation resources. The number and resolution of images in CINIC-10 are appropriate to conduct experiments. Furthermore, a fair assessment of generalization performance can be tested on CINIC-10 through the equal-sized training, validation, and testing splits.

4) Initializations: We adopt the same data augmentations as the common practices [40], [43], [47]: padding the images along each side with 4 pixels, randomly cropping from the padded images with "32×32" on CINIC-10 and CIFAR databases and horizontally flipping the images randomly. Finally, the training images will be normalized by subtracting the channel means and standard deviations. The mini-batch size is set to 128. The training epochs are set to 120 on the regular ImageNet, 200 on CINIC-10, and 400 on the CIFAR database. The SGD method and Nesterov momentum [49] are used. On the CINIC-10 datasets, the learning rate starts from 0.1 and is divided by 10 at the 100th, 150th, and 175th epochs. On the CIFAR datasets, the learning rate starts from 0.1 and decreases with the cosine learning schedule. On ImageNet datasets, the learning rate starts from 0.1 and is divided by 10 at the 30th, 60th, and 90th epochs. The momentum is 0.9, and the weight decay is set to $1e^{-4}$ on the ImageNet, CINIC-10, and CIFAR datasets.

In order to thoroughly verify the effectiveness of reducing the model size and computation FLOPs in the proposed EH-Nets, we employ five popular networks, i.e, VGG-16, ResNet-56, ResNet-110, WideResNet-28-10, and ResNet-50 as the backbone networks to implement the EH-Nets. Specifically, EH-Nets are constructed through replacing every "3×3" convolutional layer with the basic efficient harmonic block illustrated in Algorithm 1. Since the binary indices tensor $\boldsymbol{\psi}$ is discrete, it will not obtain the gradients in the backpropagation in the training process. Therefore, straight through estimator [50] is utilized to optimize binary index tensor $\boldsymbol{\psi}$. In detail, another continuous tensor $\boldsymbol{\varphi}$ is built using $\boldsymbol{\psi}$, where $\boldsymbol{\varphi}$ and $\boldsymbol{\psi}$ have the same size of "$M \times F \times m$" and $\boldsymbol{\psi}_{\cdot,i}$ is obtained through locating the maximum value $\boldsymbol{\varphi}_{\cdot,i}$ and binarizing the value with 1 in the position at the ith ($i \in \{0, 1, \ldots, F-1\}$) frequency branch. The gradients of $\boldsymbol{\psi}$ can be directly propagated to $\boldsymbol{\varphi}$ to optimize $\boldsymbol{\varphi}$ in the training process.

In the following, the EH-Nets are implemented using the backbones of VGG-16, ResNet-56, ResNet-110, WideResNet-28-10, and ResNet-50. The new networks are denoted by EH-VGG, EH-ResNet-56, EH-ResNet-110, EH-WideResNet-28-10, and EH-ResNet-50. In particular, EH-ResNet-56, EH-ResNet-110, EH-WideResNet-28-10, and EH-VGG are performed on CIFAR or CINIC-10 datasets, while EH-ResNet-50 is conducted on the ImageNet dataset. It is noticed that, in the dropping process of fusing filters, the dropping rate linearly increases from 0 to the set dropping rate in all frequency branches. Therefore, the permutation of the C-DCT filters will affect the saved meta-filters and retrieved feature in every branch. To properly arrange these frequency branches according to the increasing dropping rates, two permutation strategies for C-DCT filters are utilized in this stage. The first strategy is the frequency-based permutation. It permutes the C-DCT filters with the increasing order of frequency. The other one is the frequency-crossed permutation. It permutes the C-DCT filters row by row in the matrix of complete C-DCT filters. Finally, the initializations of these EH-Nets on various datasets are shown in Table I.

B. Ablation Study of C-DCT Filters

In this section, we will thoroughly explore the effects of different strategies utilized in the C-DCT filters. Our experiments select the EH-ResNets-56 as the backbone network on both the CIFAR-10 and CIFAR-100 datasets. The detailed initializations of the backbone networks are described in Section VI-A4. Some settings may change in the specific experiments and will be clearly pointed out in the relevant illustrations.

1) Effects of Sliding Step Size: According to Algorithm 1, the spatial size of the C-DCT filters in EH-Nets is "3×3." As the illustrations of C-DCT filters in Section III, the possible sliding step sizes λ of C-DCT filters are 1 or 2.

To study the effects of different sliding step sizes on the final performance, two EH-ResNet-56 with step sizes 1 and 2 are compared on the CIFAR-10 and CIFAR-100 datasets. The results are shown in Table II. Compared to EH-ResNet-56 with a step size of 2, EH-ResNet-56 with a step size of 1 improves the accuracy by 0.11% and 0.50% on the CIFAR-10 and CFAR-100 datasets, respectively. This proves that the proposed network with a step size of 1 can produce better accuracy on both CIFAR datasets than that with a step size of 2. Based on the integration mechanism of C-DCT filters in (4) and (5), the integrated filters with a sliding step size of 1 contain much more parameters in low frequencies than those with a step size of 2. This indicates that the feature captured in low frequencies will contribute more to the final performance than that of high frequencies in the C-DCT filters. Thus, the C-DCT

TABLE I

PROPOSED EH-NETS IMPLEMENTED USING DIFFERENT BACKBONE NETWORKS AND APPLIED TO DIFFERENT DATASETS. l AND l^* DENOTE THE SELECTION LEVELS OF UPPER TRIANGULAR AND POROUS METHODS, RESPECTIVELY. $\boldsymbol{\beta}$ INDICATES THE DROPPING RATE OF META-FILTERS. λ REPRESENTS THE SLIDING STEP SIZE OF COMBINATIONAL C-DCT FILTERS. "PERMUTATION" INDICATES THE PERMUTATION METHOD OF C-DCT FILTERS

Datasets	Models	Selection method	l / l^*	λ	$\boldsymbol{\beta}$	Permutation method
CIFAR-10	EH-ResNet-56	Upper triangular	$l = 5$	1	$\{0.2, 0.3, 0.4\}$	Frequency-based
	EH-ResNet-110	Upper triangular	$l = 5$	1	$\{0.2, 0.3, 0.4\}$	Frequency-based
	EH-WideResNet-28-10	Porous	$l^* = 5$	1	0.3	Frequency-crossed
	EH-VGG	Upper triangular	$l = 5$	1	0.3	Frequency-based
CIFAR-100	EH-ResNet-56	Porous	$l^* = 5$	1	$\{0.2, 0.3, 0.4\}$	Frequency-crossed
	EH-ResNet-110	Porous	$l^* = 5$	1	$\{0.2, 0.3, 0.4\}$	Frequency-crossed
	EH-WideResNet-28-10	Porous	$l^* = 5$	1	0.3	Frequency-crossed
	EH-VGG	Porous	$l^* = 5$	1	0.3	Frequency-crossed
CINIC-10	EH-ResNet-56	Porous	$l^* = 5$	1	$\{0.2, 0.3, 0.4\}$	Frequency-crossed
	EH-WideResNet-28-10	Porous	$l^* = 5$	1	0.3	Frequency-crossed
ImagNet	EH-ResNet-50	Porous	$l^* = 5$	1	0.3	Frequency-crossed

TABLE II

EFFECTS OF SLIDING STEP SIZE OF C-DCT FILTERS

Datasets	Models	Step size: λ	Accuracy
CIFAR-10	EH-ResNet-56	1	**93.05**%
	EH-ResNet-56	2	92.94%
CIFAR-100	EH-ResNet-56	1	**68.54**%
	EH-ResNet-56	2	68.04%

TABLE III

EFFECTS OF PERMUTATION OF C-DCT FILTERS WITH DIFFERENT SELECTION METHODS

Datasets	Models	Selection	Permutation method	Accuracy
CIFAR-10	EH-ResNet-56	Upper-triangular	frequency-based	92.25%
	EH-ResNet-56		frequency-crossed	**93.05**%
	EH-ResNet-56	Porous	frequency-based	<u>92.73</u>%
	EH-ResNet-56		frequency-crossed	92.49%
CIFAR-100	EH-ResNet-56	Upper-triangular	frequency-based	66.06%
	EH-ResNet-56		frequency-crossed	<u>68.54</u>%
	EH-ResNet-56	Porous	frequency-based	**69.06**%
	EH-ResNet-56		frequency-crossed	68.70%

filters with a sliding step size of 1 are utilized in the proposed EH-Nets.

2) Effects of Selection Methods: Two selection methods, i.e., upper triangular selection and porous selection, are utilized to select the C-DCT filters from the C-DCT filter matrix. Since the selected C-DCT filters will influence the retrieved feature from different frequencies, these two selection methods will be compared on the CIFAR-10 and CIFAR-100 datasets. Table III demonstrates the comparison results of upper triangular and porous selection methods. On CIFAR-10 datasets, the networks with upper triangular selection have an accuracy of 0.56% higher than those with porous selection method. On CIFAR-100 datasets, however, the porous selection method generates better accuracy than the upper triangular selection method.

CIFAR-10 and CIFAR-100 datasets both include the same number of training images. However, the categories of CIFAR-100 dataset are ten times larger than those of CIFAR-10 dataset. This indicates that the CIFAR-100 dataset is more challenging than the CIFAR-10 dataset. Considering the comparisons between these two selection methods on the CIFAR datasets, the produced experiment results suggest that the compound DCT filters with higher frequencies will be more necessary than those with low frequencies on difficult datasets. Therefore, in the subsequent experiments, the upper triangular selection is applied to the easy CIFAR-10 dataset, while the porous selection method is utilized on the difficult CIFAR-100, CINIC-10, and ImageNet datasets.

3) Effects of C-DCT Filter Permutation: Two permutation methods of C-DCT filters are compared on CIFAR datasets. The experimental results are shown in Table III. When using the upper triangular selection method, the networks with frequency-crossed permutation obtain 0.80% and 2.48% improvements on the CIFAR-10 and CIFAR-100 datasets, respectively, compared to the networks with the frequency-based permutation. However, when using the porous selection method, the networks with the frequency-based permutation obtain 0.24% and 0.36% improvements on CIFAR-10 and CIFAR-100 datasets, respectively, compared to the networks with frequency-crossed permutation. This is mainly because the upper triangular selection collects much more C-DCT filters from low frequencies compared to the porous selection method. This will lead to overfitting in the networks under the upper triangular selection method. Furthermore, compared to the frequency-based permutation, the frequency-crossed permutation will drop more meta-filters in low-frequency branches under upper triangular selection method. Therefore, the integration of upper triangular selection and frequency-crossed permutation can achieve better accuracy. The porous selection method can collect only a few C-DCT filters in low frequencies. Combining the frequency-based permutation with porous selection method, the meta-filters will be largely saved in the low-frequency branches and thus avoid the overfitting and produce better performance to some extent.

TABLE IV

EFFECTS OF REDUCTION FACTOR α ON THE MODEL SIZE (#PARAM.), COMPUTATION FLOPS (#FLOPS), AND ACCURACY. THE COMPLETE C-DCT FILTERS, I.E., 25 C-DCT FILTERS [SEE (4) AND (5)], ARE USED IN THE BACKBONE EH-RESNET-56. THE STRATEGY OF DROPPING META-FILTERS IS REMOVED IN THIS EXPERIMENT

Datasets	Models	α	#Param.	#FLOPs	Accuracy
CIFAR-10	EH-ResNet-56	0.25	$26.8K$	$0.21G$	92.65%
	EH-ResNet-56	0.50	$50.3K$	$0.30G$	**93.57%**
	EH-ResNet-56	0.75	$73.9K$	$0.39G$	93.23%
CIFAR-100	EH-ResNet-56	0.25	$32.5K$	$0.21G$	69.15%
	EH-ResNet-56	0.50	$56.1K$	$0.30G$	70.28%
	EH-ResNet-56	0.75	$79.6K$	$0.39G$	**71.03%**

TABLE V

EFFECTS ON COMPUTATION FLOPS (#FLOPS) AND ACCURACY OF SELECTION LEVEL l/l^* AND DROP RATES $\boldsymbol{\beta}$ ON THREE STAGES OF NETWORKS. "$-$" IN THE COLUMN l/l^* INDICATES THE NETWORKS UTILIZING COMPLETE C-DCT FILTERS, I.E., 25 C-DCT FILTERS [SEE (4) AND (5)]

Datasets	Models	l/l^*	$\boldsymbol{\beta}$	#FLOPs	Accuracy
CIFAR-10	EH-ResNet-56	$-$	$\{0.2, 0.2, 0.2\}$	$0.29G$	93.12%
	EH-ResNet-56	$-$	$\{0.4, 0.4, 0.4\}$	$0.27G$	93.41%
	EH-ResNet-56	$-$	$\{0.5, 0.5, 0.5\}$	$0.26G$	93.30%
	EH-ResNet-56	$-$	$\{0.2, 0.3, 0.4\}$	$0.28G$	93.15%
	EH-ResNet-56	$l = 5$	$\{0.4, 0.4, 0.4\}$	$0.16G$	92.80%
	EH-ResNet-56	$l = 5$	$\{0.2, 0.3, 0.4\}$	$\mathbf{0.17G}$	**93.05%**
CIFAR-100	EH-ResNet-56	$l = 5$	$\{0.4, 0.4, 0.4\}$	$0.16G$	68.54%
	EH-ResNet-56	$l = 5$	$\{0.2, 0.3, 0.4\}$	$\mathbf{0.17G}$	**69.09%**

TABLE VI

EFFECTS ON ACCURACY OF C-DCT FILTERS. THE DEFAULT SETTINGS OF THESE NETWORKS ON DIFFERENT DATASETS ARE SHOWN IN TABLE I

Methods	Filters	CIFAR-10	CIFAR-100	CINIC-10	ImageNet
EH-ResNet-56	DCT	91.51%	68.66%	79.95%	–
	C-DCT	**93.05%**	**69.06%**	**80.31%**	–
EH-ResNet-110	DCT	92.92%	68.83%	–	–
	C-DCT	**93.19%**	**70.42%**	–	–
EH-VGG-16	DCT	92.76%	65.93%	–	–
	C-DCT	**92.87%**	**68.78%**	–	–
EH-WideResNet-28-10	DCT	95.92%	79.11%	82.37%	–
	C-DCT	**96.16%**	**81.51%**	**84.42%**	–
EH-ResNet-50	DCT	–	–	–	76.35%
	C-DCT	–	–	–	**76.83%**

In summary, the experimental results of comparisons above have demonstrated the initial settings of the networks on the CIFAR-10 and CIFAR-100 datasets. Since the CINIC-10 and ImageNet datasets are both much more difficult datasets than the CIFAR-10 datasets, the initial settings of the networks on these two datasets are the same as those on the CIFAR-100 dataset.

C. Ablation Study of Shared Reconstruction Filter

This section will comprehensively study the effects of different settings of shared reconstruction filters in terms of model size, computation FLOPs, and accuracy.

1) Effects of Reduction Factor α: In the shared reconstruction filters, the reduction factor α for meta-filters will heavily affect the model size and computation FLOPs. Table IV shows the results of different numbers of meta-filters in EH-ResNets-56. From this table, the model size and computational FLOPs increase with α. On the CIFAR-100 dataset, a larger value of α leads to higher accuracy. However, on the CIFAR-10 dataset, the accuracy increases and then decreases. The networks with the largest model size ($\alpha = 0.75$) produce lower accuracy than those with the middle model size ($\alpha = 0.5$) on the CIFAR-10 dataset. Considering the overall effects of the model size, computation FLOPs, and accuracy on different datasets, $\alpha = 0.5$ is set as the initial settings for the proposed EH-Nets.

2) Effects of Selection Level l/l^* and Dropping Rate β: According to the calculations of parameters and FLOPs in (5) and (19), the selection level l/l^* in the C-DCT filters and drop rate $\boldsymbol{\beta}$ for the meta-filters in the shared reconstruction filters will affect only the computation FLOPs. Therefore, the combined factors of selection level and dropping rate will be explored together to discover their effects on the FLOPs and accuracy. Table V shows the comparison results of different settings in EH-ResNet-56 on the CIFAR-10 and CIFAR-100 datasets.

As can be seen, with the increase of computation FLOPs, the accuracy of the networks increases and then decreases on CIFAR-10. This is consistent with the accuracy trend of the model size in Table IV. Since the selection level and dropping rates do not affect the model size, the model size in Table V is all the same. Hence, according to the comparative results, the computational FLOPs may also produce an important effect on the final performance. This also implies that the overfitting problem possibly results from too many computational FLOPs on easy dataset. On CIFAR-100, the networks with $0.17G$ FLOPs can improve the accuracy by 0.55% compared to the networks with $0.16G$ FLOPs. Thus, increasing only a few computation FLOPs, the accuracy can be promoted satisfactorily on difficult datasets without increasing the model size. This observation can provide novel guidance to design lightweight and effective networks. Finally, according to the comprehensive comparisons from both FLOPs and accuracy on CIFAR datasets, $l/l^* = 5$ and $\boldsymbol{\beta} = \{0.2, 0.3, 0.4\}$ are utilized in the subsequent ResNet family networks.

D. Effect of C-DCT Filters

In order to verify the effectiveness of C-DCT filters, we compare the performances of various EH-Nets with C-DCT filters and with the traditional DCT filters. The experiment results on CIFAR, CINIC-10, and ImageNet datasets are shown in Table VI. From this table, compared to DCT filters, EH-ResNet-56 with C-DCT filters improves the accuracies by 1.54%, 0.40%, and 0.36% on the CIFAR-10, CIFAR-100, and CINIC-10 datasets, respectively. EH-ResNet-110 produces 0.27% and 1.59% higher accuracies than the networks with traditional DCT filters. On the backbone VGG-16 networks,

TABLE VII

COMPARISONS OF MODEL SIZE (#PARAMS.), COMPUTATION FLOPS (#FLOPS), AND ACCURACY. THE DEFAULT SETTINGS OF THESE NETWORKS ON DIFFERENT DATASETS ARE SHOWN IN TABLE I. † AND ‡ DENOTE USING THE COMPLETE C-DCT FILTERS AND REMOVING THE DROP STRATEGY FOR THE SHARED META-FILTERS, RESPECTIVELY

Methods	#Params.	Reduction	#FLOPs	Reduction	CIFAR-10	CIFAR-100	CINIC-10	ImageNet (Top-1)
ResNet-56	$0.85M$	$1.0\times$	$0.13G$	$1.0\times$	93.03%	–		
Harm-ResNet-56	$0.85M$	$1.0\times$	$0.13G$	$1.0\times$	93.21%	$\mathbf{71.81\%}$	–	–
EH-ResNet-56 (ours)	$\mathbf{0.05M}$	$\mathbf{17.0\times}$	$0.17G$	$0.76\times$	93.05%	69.06%		
EH-ResNet-56†‡ (ours)	$\mathbf{0.05M}$	$\mathbf{17.0\times}$	$0.30G$	$0.43\times$	$\mathbf{93.57\%}$	70.28%		
ResNet-110	$1.7M$	$1.0\times$	$0.26G$	$1.0\times$	93.39%	–		
Harm-ResNet-110	$1.7M$	$1.0\times$	$0.26G$	$1.0\times$	93.81%	$\mathbf{72.18\%}$	–	–
EH-ResNet-110 (ours)	$\mathbf{0.1M}$	$\mathbf{17.0\times}$	$0.34G$	$0.76\times$	93.19%	70.42%		
EH-ResNet-110†‡ (ours)	$\mathbf{0.1M}$	$\mathbf{17.0\times}$	$0.60G$	$0.43\times$	$\mathbf{93.85\%}$	71.58%		
WideResNet-28-10	$36.5M$	$1.0\times$	$5.24G$	$1.0\times$	96.00%	80.75%	84.16%	
Harm-WideResNet-28-10	$36.5M$	$1.0\times$	$5.24G$	$1.0\times$	96.14%	81.43%	84.38%	–
EH-WideResNet-28-10 (ours)	$\mathbf{2.3M}$	$\mathbf{15.8\times}$	$\mathbf{3.37G}$	$\mathbf{1.6\times}$	$\mathbf{96.16\%}$	$\mathbf{81.51\%}$	$\mathbf{84.42\%}$	
ResNet-50	$25.6M$	$1.0\times$	$3.8G$	$1.0\times$				76.19%
Harm-ResNet-50	$25.6M$	$1.0\times$	$3.8G$	$1.0\times$	–	–	–	$\mathbf{76.89\%}$
EH-ResNet-50 (ours)	$\mathbf{14.7M}$	$\mathbf{1.7\times}$	$\mathbf{3.2G}$	$\mathbf{1.2\times}$				76.83%

the accuracies produced from EH-VGG with C-DCT filters are, respectively, promoted by 0.11% and 2.85% on the CIFAR-10 and CIFAR-100 datasets. On the backbone WideResNet-28-10, the networks with C-DCT filters can improve the accuracies by 0.24%, 2.40%, and 2.05 on the CIFAR-10, CIFAR-100, and CINIC-10 datasets, respectively. ResNet-50 is suitable for the ImageNet dataset and the EH-ResNet-50 with C-DCT filters can produce 76.83% accuracy on the ImageNet dataset, improving the accuracy by 0.48% compared to that with DCT filters. From these comparisons, on various backbone structures, the EH-Nets with C-DCT filters can achieve higher accuracies on all datasets compared to those with traditional DCT filters. This sufficiently demonstrates that the networks with C-DCT filters can drive the networks to learn more qualified features, resulting in better performances.

To explicitly prove the function of C-DCT filters, we visualize the importance of C-DCT filters and DCT filters through statistics of the average absolute weights from the shared reconstruction filters corresponding to the relevant frequency filters. Fig. 6(a) shows the comparisons at the first layer and the last layer in three stages from EH-ResNet-56 on the CIFAR-10 dataset. It is clear that the importance of some C-DCT filters is significantly higher than that of the DCT filters, such as the importance marked by blue boxes. In these four layers, the highest importance is produced from C-DCT filters, demonstrating the effectiveness of C-DCT filters.

Furthermore, in EH-ResNet-56, the average importance of C-DCT filters and DCT filters in every layer is also calculated and shown in Fig. 6(b). According to the comparisons in this figure, in most of the layers (31/55), the average importance of C-DCT filters is higher than that of DCT filters. This further proves that the C-DCT filters play an important role in retrieving helpful feature for the entire networks, contributing to better performances.

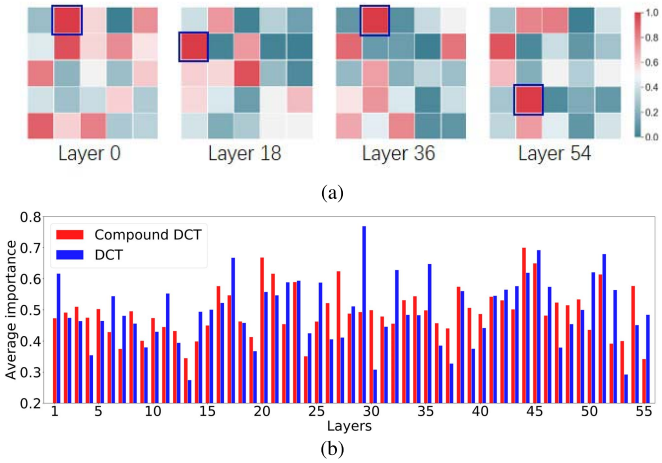

Fig. 6. Importance comparisons of C-DCT filters and DCT filters in EH-ResNet-56 on the CIFAR-10 dataset. (a) Comparisons of importance of all the filters. (b) Comparisons of average importance of every layer.

E. Comprehensive Comparisons

To verify the effectiveness of the proposed EH-Nets, we comprehensively compare the model size, FLOPs, and accuracy on various datasets and backbone networks in this section. Table VII shows the final experiment results. On both CIFAR-10 and CIFAR-100 datasets, the model size of EH-ResNet-56 and EH-ResNet-110 can be significantly reduced by 17 times than the Harm-ResNet-56 and Harm-ResNet-110. According to (19), since the widths of ResNet-56 and ResNet-110 are very small, the calculations of FLOPs will slightly increase in the proposed EH-ResNet-56 and EH-ResNet-110 networks. The accuracies, however, achieved by EH-ResNet-56 and EH-ResNet-110 are higher than those of Harm-ResNet-56 and Harm-ResNet-110 on the CIFAR-10 dataset. Although the model size is largely reduced, the abundant C-DCT filters and increasing of FLOPs can improve the quality of retrieved feature, leading to better performance on easy datasets. However, on the challenging CIFAR-100 datasets, the accuracies produced by the proposed networks are decreased slightly. This is because the very small width of the networks can limit the richness of the shared reconstruction filters, leading to insufficient fusion of the features from various frequency branches.

TABLE VIII

COMPARISONS TO THE STATE-OF-THE-ART MOBILE CNNS ON CIFAR DATASETS. † AND ‡ DENOTE USING THE COMPLETE C-DCT FILTERS AND REMOVING THE DROP STRATEGY FOR THE SHARED META-FILTERS, RESPECTIVELY

Methods	#Params.	#FLOPs	CIFAR-10	CIFAR-100
MobileNetV2 1.0× [18]	$2.3M$	$0.11G$	94.56%	77.09%
IGCV3-D 1.0× [22]	$2.3M$	$0.10G$	94.96%	77.95%
ShuffleNetV2 2.0× [24]	$5.5M$	$0.17G$	93.77%	75.17%
Harm-ResNet-56	$0.85M$	$0.13G$	93.21%	71.81%
EH-ResNet-56†‡ (ours)	$\mathbf{0.05M}$	$0.30G$	93.57%	70.28%
Harm-WideResNet-28-10	$36.5M$	$5.24G$	96.14%	81.43%
EH-WideResNet-28-10 (ours)	$2.3M$	$3.37G$	**96.16%**	**81.51%**

Based on these comparisons on ResNet-56 and ResNet-110, as the experiments in the literature [8], EH-WideResNet-28-10 with a lager width is implemented. EH-WideResNet-28-10 reduces the model size and FLOPs by 15.8 and 1.6 times, respectively, but produces higher accuracies on the CIFAR-10, CIFAR-100, and CINIC-10 datasets. Thus, on the relatively wider networks, the proposed method can reduce both the model size and computation FLOPs efficiently without any performance degradation. On the ImageNet dataset, the proposed EH-ResNet-50 can achieve competitive accuracy while reducing the model size and computation FLOPs by 1.7 times and 1.2 times, respectively. From all the above comparisons, the proposed method can effectively reduce the number of parameters and computation FLOPs while maintaining satisfactory performances on various networks.

In addition, because the implementation of the proposed shared reconstruction filters has some relationships of the compression methods using group convolution, we compare our EH-Nets to the popular state-of-the-art mobile CNNs with group convolution. Table VIII shows the comparisons of parameters, FLOPs, and accuracies on the CIFAR-10 and CIFAR-100 datasets. The compared mobile CNNs are the popular MobileNetV2, ShuffleNetV2, and IGCV3. From Table VIII, compared to the mobile networks, our EH-ResNet-56†‡ significantly reduces the parameters by about 46 times while producing competitive accuracies on the CIFAR-10 dataset. Compared to the mobile networks under the same parameters, our EH-WideResNet-28-10 achieves the best accuracies on both CIFAR-10 and CIFAR-100 datasets. Due to the additional computations from the C-DCT, the proposed EH-Nets need more FLOPs in the learning process. However, compared to the traditional HNNs (Harm-WideResNet-28-10), our EH-Nets (EH-WideResNet-28-10) can decrease the FLOPs by $1.87G$. In summary, compared to the mobile CNNs under the same model size, our EH-Nets can effectively improve the accuracy. Compared to the HNNs, the proposed EH-Nets can effectively reduce both the model size and FLOPs while significantly improving the accuracy. The comparisons to the HNNs prove the effectiveness of the proposed EH-Nets, which is also consistent with our goal of improving the performance and efficiency for HNNs.

In order to explicitly compare the proposed EH-Net and the HNNs, the important calculations of the DCT filters from EH-WideResNet-28 and Harm-WideResNet-28 are compared in Fig. 7. In this figure, the larger importance of DCT filters in Harm-WideResNet-28 mainly focuses on the low-frequency DCT filters, while our EH-WideResNet-28 produces larger importance on some compound DCT-filters even on some high-frequency C-DCT filters. This proves that our method can learn more appropriate combinations of necessary C-DCT filters, mainly because the dropping strategy in the shared recombination filters can learn and dynamically select the required C-DCT filters. Therefore, our EH-Nets can effectively explore the richness of the C-DCT filters with limited parameters, leading to better promotions of performance and efficiency.

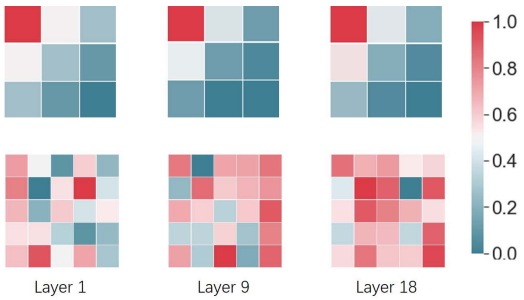

Fig. 7. Importance comparisons of DCT filters and C-DCT filters in Harm-WideResNet-28 and Harm-WideResNet-28 (ours) on the CIFAR-10 dataset, respectively. The top column and the bottom column are the importances from DCT filters and C-DCT filters, respectively.

VII. CONCLUSION

This article proposed EH-Nets. This is the first attempt to compress harmonic networks while maintaining the networks' performance. EH-Nets are mainly constructed by two proposed filters, including C-DCT filter and shared reconstruction filter. The C-DCT filters are generated by integrating the nearest DCT filters to retrieve the rich spectrum features. The shared reconstruction filters are stacked with selected meta-filters. Since the number of meta-filters is much smaller than that of traditional fusing filters in traditional harmonic networks, this can significantly reduce the model size of networks. Furthermore, in order to reduce the computation complexity, the meta-filters can be dropped in some frequency branches. In the calculation process, the meta-filters are performed in every frequency branch to share the computations. Through the dropping strategy and sharing computations of meta-filters, the computation complexity can be greatly reduced in the networks. Experiment results proved that the proposed EH-Nets can effectively reduce the model size and computation FLOPs of various networks with a regular width while preserving performance well on extensive datasets.

REFERENCES

[1] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2261–2269.

[2] Y. Lu, G. Lu, B. Zhang, Y. Xu, and J. Li, "Super sparse convolutional neural networks," in *Proc. 33rd AAAI Conf. Artif. Intell. (AAAI)*, vol. 33, 2019, pp. 4440–4447.

[3] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.

[4] Q. He, X. Sun, Z. Yan, and K. Fu, "DABNet: Deformable contextual and boundary-weighted network for cloud detection in remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–16, 2022.
[5] C. Yan, B. Gong, Y. Wei, and Y. Gao, "Deep multi-view enhancement hashing for image retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 4, pp. 1445–1451, Apr. 2020.
[6] C. Yan, Z. Li, Y. Zhang, Y. Liu, X. Ji, and Y. Zhang, "Depth image denoising using nuclear norm and learning graph model," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 16, no. 4, pp. 1–17, Nov. 2020.
[7] C. Yan *et al.*, "Task-adaptive attention for image captioning," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 1, pp. 43–51, Jan. 2022.
[8] M. Ulicny, V. A. Krylov, and R. Dahyot, "Harmonic convolutional networks based on discrete cosine transform," 2020, *arXiv:2001.06570*.
[9] P. V. S. Ponnapalli, K. C. Ho, and M. Thomson, "A formal selection and pruning algorithm for feedforward artificial neural network optimization," *IEEE Trans. Neural Netw.*, vol. 10, no. 4, pp. 964–968, Jul. 1999.
[10] X. Ruan *et al.*, "EDP: An efficient decomposition and pruning scheme for convolutional neural network compression," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 10, pp. 4499–4513, Oct. 2021.
[11] T. Zhang *et al.*, "StructADMM: Achieving ultrahigh efficiency in structured pruning for DNNs," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 5, pp. 2259–2273, May 2022.
[12] G. Huang, S. Liu, L. van der Maaten, and K. Q. Weinberger, "CondenseNet: An efficient densenet using learned group convolutions," 2017, *arXiv:1711.09224*.
[13] Z. Chen, T.-B. Xu, C. Du, C.-L. Liu, and H. He, "Dynamical channel pruning by conditional accuracy change for deep neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 2, pp. 799–813, Feb. 2021.
[14] I.-C. Lin *et al.*, "A novel, efficient implementation of a local binary convolutional neural network," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 68, no. 4, pp. 1413–1417, Apr. 2021.
[15] A. Bulat and G. Tzimiropoulos, "Hierarchical binary CNNs for landmark localization with limited resources," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 343–356, Feb. 2020.
[16] H. Peng, J. Wu, Z. Zhang, S. Chen, and H.-T. Zhang, "Deep network quantization via error compensation," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Apr. 14, 2021, doi: 10.1109/TNNLS.2021.3064293.
[17] A. G. Howard *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.
[18] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation," 2018, *arXiv:1801.04381*.
[19] D. Zhou, Q. Hou, Y. Chen, J. Feng, and S. Yan, "Rethinking bottleneck structure for efficient mobile network design," 2020, *arXiv:2007.02269*.
[20] T. Zhang, G.-J. Qi, B. Xiao, and J. Wang, "Interleaved group convolutions," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 4383–4392.
[21] G. Xie, J. Wang, T. Zhang, J. Lai, R. Hong, and G.-J. Qi, "Interleaved structured sparse convolutional neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8847–8856.
[22] K. Sun, M. Li, D. Liu, and J. Wang, "IGCV3: Interleaved low-rank group convolutions for efficient deep neural networks," 2018, *arXiv:1806.00178*.
[23] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," 2017, *arXiv:1707.01083*.
[24] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "ShuffleNet v2: Practical guidelines for efficient CNN architecture design," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds., 2018, pp. 122–138.
[25] Y. Lu, G. Lu, R. Lin, J. Li, and D. Zhang, "SRGC-Nets: Sparse repeated group convolutional neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 8, pp. 2889–2902, Aug. 2020.
[26] Y. Lu, G. Lu, Y. Zhou, J. Li, Y. Xu, and D. Zhang, "Highly shared convolutional neural networks," *Expert Syst. Appl.*, vol. 175, Aug. 2021, Art. no. 114782.
[27] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," 2017, *arXiv:1707.07012*.
[28] C. Liu *et al.*, "Progressive neural architecture search," 2017, *arXiv:1712.00559*.
[29] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Jul. 2019, pp. 4780–4789.
[30] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.
[31] M. Zhu, J. Li, N. Wang, and X. Gao, "Knowledge distillation for face photo-sketch synthesis," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 2, pp. 893–906, Feb. 2022.
[32] Q. Zhao, J. Dong, H. Yu, and S. Chen, "Distilling ordinal relation and dark knowledge for facial age estimation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 7, pp. 3108–3121, Jul. 2021.
[33] R. C. Gonzalez *et al.*, *Digital Image Processing*. Upper Saddle River, NJ, USA: Prentice-Hall, 2002.
[34] A. Raid, W. Khedr, M. El-dosuky, and W. Ahmed, "Jpeg image compression using discrete cosine transform—A survey," *Int. J. Comput. Sci. Eng. Surv.*, vol. 5, no. 2, p. 39, 2014.
[35] F. Franzen, "Image classification in the frequency domain with neural networks and absolute value DCT," in *Proc. Int. Conf. Image Signal Process.*, Cham, Switzerland: Springer, 2018, pp. 301–309.
[36] S. F. D. Santos, N. Sebe, and J. Almeida, "The good, the bad, and the ugly: Neural networks straight from JPEG," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2020, pp. 1896–1900.
[37] L. Gueguen, A. Sergeev, B. Kadlec, R. Liu, and J. Yosinski, "Faster neural networks straight from jpeg," in *Proc. Adv. Neural Inf. Process. Syst.*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Red Hook, NY, USA: Curran Associates, 2018, pp. 3937–3948.
[38] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," 2016, *arXiv:1608.08710*.
[39] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[40] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[41] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, R. C. Wilson, E. R. Hancock, and W. A. P. Smith, Eds., Sep. 2016, pp. 87.1–87.12.
[42] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.
[43] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Weinberger, "Deep networks with stochastic depth," in *Eur. Conf. Comput. Vis. (ECCV)*, Oct. 2016, pp. 646–661.
[44] S. Xie, R. Girshick, P. Dollar, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2017, pp. 5987–5995.
[45] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, U.K., Tech. Rep., 2009.
[46] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: A large data set for nonparametric object and scene recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 11, pp. 1958–1970, Nov. 2008.
[47] G. Larsson, M. Maire, and G. Shakhnarovich, "FractalNet: Ultra-deep neural networks without residuals," 2016, *arXiv:1605.07648*.
[48] L. N. Darlow, E. J. Crowley, A. Antoniou, and A. J. Storkey, "CINIC-10 is not ImageNet or CIFAR-10," 2018, *arXiv:1810.03505*.
[49] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2013, pp. 1139–1147.
[50] M. Courbariaux, Y. Bengio, and J.-P. David, "BinaryConnect: Training deep neural networks with binary weights during propagations," in *Proc. 28th Int. Conf. Neural Inf. Process. Syst.*, vol. 2. Cambridge, MA, USA: MIT Press, Dec. 2015, pp. 3123–3131.

Yao Lu received the B.S. degree in computer science and technology from Huaqiao University, Xiamen, China, in 2015, and the Ph.D. degree in computer applied technology from the Harbin Institute of Technology, Harbin, China, in 2020.

She was a Post-Doctoral Fellow with the University of Macau, Macau, China, from 2020 to 2021. She is currently an Assistant Professor with the Biocomputing Research Center, Harbin Institute of Technology, Shenzhen, China. Her research interests include pattern recognition, deep learning, computer vision, and relevant applications.

Le Zhang received the B.S. degree in communication engineering from the Renai College, Tianjin University, Tianjin, China, in 2016, and the M.S. degree in information and communication engineering from Yanshan University, Qinhuangdao, China, in 2019. She is currently pursuing the Ph.D. degree with the Harbin Institute of Technology (Shenzhen), Shenzhen, China.

Her current research interests include pattern recognition, deep learning, computer vision, and image steganography.

Xiaofei Yang received the B.S. degree from Suihua University, Suihua, China, in 2011, and the M.S. and Ph.D. degrees from the Harbin Institute of Technology, Harbin, China, in 2014 and 2019, respectively.

He currently holds a post-doctoral position at the Department of Computer and Information Science, University of Macau, Macau, China. His research interests are in the areas of semisupervised learning, deep learning, remote sensing, transfer learning, and graph mining.

Yicong Zhou (Senior Member, IEEE) received the B.S. degree in electrical engineering from Hunan University, Changsha, China, in 1992, and the M.S. and Ph.D. degrees in electrical engineering from Tufts University, Medford, MA, USA, in 2008 and 2010, respectively.

He is currently a Professor with the Department of Computer and Information Science, University of Macau, Macau, China. His research interests include image processing, computer vision, machine learning, and multimedia security.

Dr. Zhou is a fellow of the Society of Photo-Optical Instrumentation Engineers (SPIE). He was recognized as one of "Highly Cited Researchers" in 2020 and 2021. He received the Third Price of Macao Natural Science Award as a sole winner in 2020 and was a co-recipient in 2014. He has been the leading Co-Chair of the Technical Committee on Cognitive Computing in the IEEE Systems, Man, and Cybernetics Society since 2015. He serves as an Associate Editor for IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON NEUTRAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, and IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING.