Expert Systems With Applications 175 (2021) 114782

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Highly shared Convolutional Neural Networks

Yao Lu [b,a,1], Guangming Lu [a,*,2], Yicong Zhou [b,3], Jinxing Li [a,4], Yuanrong Xu [c,5], David Zhang [c,d,6]

[a] Harbin Institute of Technology, Shenzhen, China
[b] University of Macau, Macau, China
[c] The Chinese University of Hong Kong, Shenzhen, China
[d] Shenzhen Research Institute of Big Data, China

ARTICLE INFO

Keywords:
Deep learning
CNNs
Group convolutions
Highly shared convolutions
HSC-Nets

ABSTRACT

In order to deploy deep Convolutional Neural Networks (CNNs) on the mobile devices, many mobile CNNs are introduced. Currently, some online applications are usually re-trained because of the constantly-increasing data. However, compared with the regular models, it is not very efficient to train the present mobile models. Therefore, the purpose of this paper is to propose efficient mobile models both in the training and test processes through exploring the main causes of the current mobile CNNs' inefficiency and the parameters' properties. Finally, this paper introduces Highly Shared Convolutional Neural Networks (HSC-Nets). The HSC-Nets employ two shared mechanisms to reuse the filters comprehensively. Experimental results showed that, compared with the regular networks and the latest state-of-the-art group-conv mobile networks, the HSC-Nets can achieve promising performances and effectively decrease the model size. Furthermore, it is also more efficient in both the training and test processes.

1. Introduction and related work

Nowadays, many mobile Convolutional Neural Networks (CNNs) are proposed to be deployed on mobile devices. The present popular mobile networks can be classified into three categories.

(a) Automatic networks structure search methods, *e.g.*, the well-known NASNet family methods (Zoph, Vasudevan, Shlens, & Le, 2018; Liu et al., 2018; Real, Aggarwal, Huang, & Le, 2018). Many optimal mobile networks can be found by these searching methods. The final obtained networks can perform outstandingly. However, since some practical applications usually need re-training with the increasing of the data information, compared with the regular models' training cost, it is more complex, memory cost and inflexible in the searching process. Furthermore, as the illustrations in Ma, Zhang, Zheng, and Sun (2018), the final searched mobile networks usually have many fragments, also resulting in time consuming in the test process. Therefore, it is not very efficient for the online applications.

(b) Pruning or dynamic pruning methods. The pruning methods, such as Lauret, Fock, and Mara (2006), Wang, Xu, Yang, and Zurada (2018), Lin, Rao, Lu, and Zhou (2017), Dong, Chen, and Pan (2017), He, Zhang, and Sun (2017), Luo, Wu, and Lin (2017), Yang, Chen, and Sze (2018) and Hu, Sun, Li, Wang, and Gu (2018), are abundant in different pruning levels or strategies. Although the final pruned models are light-weighted and efficient, they still can not be trained effectively. Because in the training process, the pruning methods should be fine-tuned when the new connections, neurons, filters or channels are dropped. Accordingly, this kind of methods are also more complex, inflexible and time consuming in the training process. Based on the pruning methods, dynamic pruning methods are introduced. One of the most popular methods is CondenseNet (Huang, Liu, Maaten, & Weinberger, 2018), which can dynamically remove the connections in the training process. And, the obtained mobile networks can achieve better performances.

* Corresponding author.
E-mail addresses: yaolu@um.edu.mo (Y. Lu), luguangm@hit.edu.cn (G. Lu), yicongzhou@um.edu.mo (Y. Zhou), csdzhang@comp.polyu.edu.hk (D. Zhang).
[1] ORCID ID: https://orcid.org/0000-0002-3147-2081
[2] ORCID ID: https://orcid.org/0000-0003-1578-2634
[3] ORCID ID: https://orcid.org/0000-0002-4487-6384
[4] ORCID ID: https://orcid.org/0000-0001-5156-0305
[5] ORCID ID: https://orcid.org/0000-0002-9457-7956
[6] ORCID ID: https://orcid.org/0000-0002-5027-5286

https://doi.org/10.1016/j.eswa.2021.114782
Received 24 June 2019; Received in revised form 13 January 2021; Accepted 22 February 2021
Available online 4 March 2021

However, as demonstrated in Ma et al. (2018), it can not be effectively implemented currently. Therefore, these methods are still not practical enough for the mobile devices.

(c) Group-conv mobile networks. Group convolutions are proposed to remove the redundancy of the regular convolutional filters from the channel extent. It initially divides the input channels and the convolutional filters into a specific number of groups equally, and only performs the corresponding convolutions in every group. Specially, when the number of the groups is equal to one, it is the regular convolutions, and when the groups are equal to the channels, it will become "depth-wise" convolutions. In a regular convolutional kernel, the value of the spatial size is always much smaller than that of the channels (*e.g.*, a convolutional kernel with size "$3\times 3\times 128\times 128$"). Consequently, group convolutions are utilized to reduce the channel extent's redundancy. The comparisons of regular and group convolutions are shown in the Fig. 1(a) and (b). Group convolutions are widely employed in the MobileNet family (Howard et al., 2017; Sandler, Howard, Zhu, Zhmoginov, & Chen, 2018), IGCV family (Zhang, Qi, Xiao, & Wang, 2017; Xie et al., 2018; Sun, Li, Liu, & Wang, 2018) and ShuffleNet family (Zhang, Zhou, Lin, & Sun, 2018; Ma et al., 2018) networks. Additionally, the work of ChannelNets (Gao, Wang, Cai, & Ji, 2020) proposes the concept of "channel-wise" convolutions. The "channel-wise" convolutions use shared 1-D ("1×1") convolutions at the channel extent to reduce the parameters and computational complexity. ChannelNets further propose group "channel-wise" convolutions and "depth-wise" separable channel-wise convolutions to construct networks. The basic operations of these two versions of "channel-wise" convolutions are also group convolution and "depth-wise" convolution, respectively.

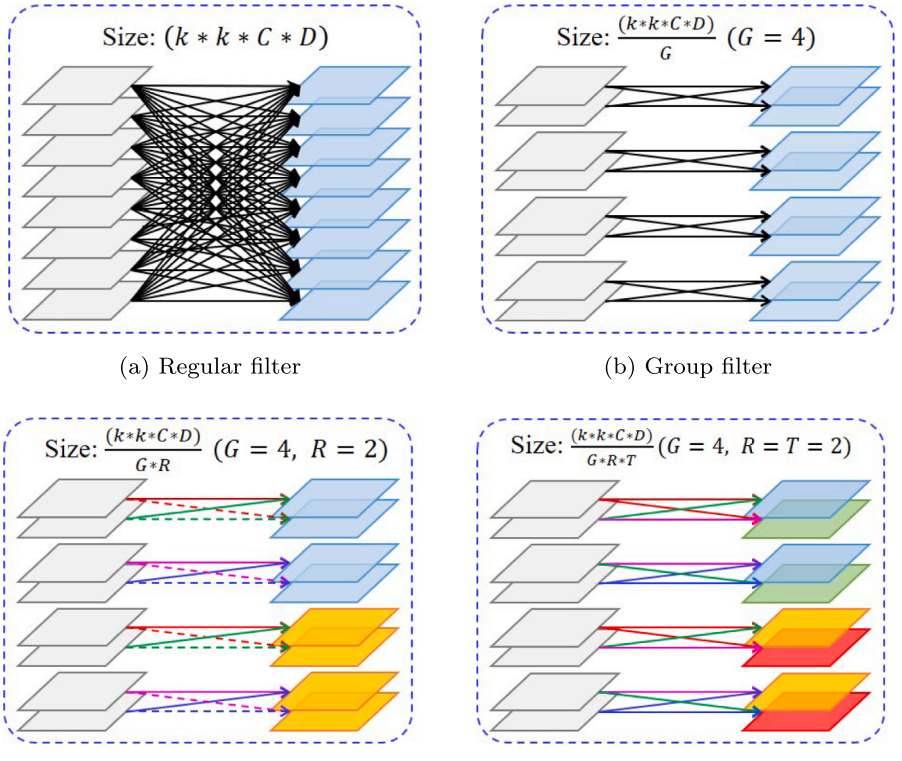
Compared with the first two kinds of methods, it is easier and more flexible to train the group-conv mobile networks. Furthermore, the training processes do not need very large computing resources. Accordingly, this paper will follow the group-conv networks to design more efficient models. In the following sections, the main causes of the network's inefficiency and the parameters' properties in the network will be explored. Based on these analysis, the principles and motivations of designing mobile models will be illustrated. Finally, we will propose our methods.

2. Further analysis and motivations

2.1. Effects of the group convolutions

As illustrated in Ma et al. (2018), the number of Flops can not properly evaluate the models' efficiency. The training and test runtime can be affected by the implementations in practice. Since the "group convolutions" can not be well supported by the deep-learning platforms (Huang et al., 2018), although it can reduce the model size and Flops, it is still very time consuming. This suggests that only increasing the number of groups in group convolutions to reduce the flops may not decrease the runtime in practice, especially for the "depth-wise" convolutions.

From another aspect, in "group convolutions" (Fig. 1(b)), one filter can only retrieve a small part of input channels, leading to poor generalization and representations, especially for the "depth-wise" convolutions. Therefore, much more epoches are required to ensure enough information to be processed by the "group convolutions" to preserve the parameters' generalization. For instance, the MobileNet, IGCV and ShuffleNet usually need much more epoches to train the networks compared with the regular models (*e.g.*, ResNet: 120 epoches (He, Zhang, Ren, & Sun, 2016); IGCV: 480 epoches (Sun et al., 2018); ShuffleNet: about 250 epoches (Ma et al., 2018) conducted on the ImageNet datasets), because the "depth-wise" convolutions are largely used in



(a) Regular filter (b) Group filter

(c) Channel repeated filter (d) Channel repeated filter with arranged shared mechanism

Fig. 1. Comparisons of the regular convolution, group convolution and their various versions with different shared mechanisms. "$k*k$" indicates the spatial size of filter. C and D represent the number of input and output channels, respectively. G, R and T respectively represent the number of groups, channel repeated times and the arranged sharing times. In (a) and (b), every black line indicates a plane kernel and these kernels are all different. In (c) and (d), the lines with same color and style are the shared plane filters. The feature maps colored blue are obtained without using any shared methods. The feature maps colored yellow are produced by channel repeated shared method. And the green feature maps use the arranged shared method. The red feature maps apply both the two sh.ared mechanisms.

them. This will further bring much more training time costs. Accordingly, the "depth-wise" convolutions can improve both the training time and the test runtime in the group-conv models. This implies the current group-conv mobile networks can further improve the efficiency by avoiding employing the "depth-wise" convolutions.

2.2. Design principles of the mobile networks

Based on the previous analysis in Section 2.1, the principles of designing mobile models can be formulated: (1) Declining the number of groups moderately in group convolutions may achieve a better trade-off between the Flops and practical runtime. (2) In order to improve the generalization of the group convolutions, it is critical to force every filter to retrieve more information in one training epoch, contributing to no more training epoches added and decreasing the training time.

However, according to the first principle, if the "depth-wise" convolutions are not used in the models, the number of the parameters will be increased, leading to the inefficiency resulting from the improvement of memory access cost (MAC). Because as the demonstrations in the Ma et al. (2018), the MAC contains the memory cost of the parameters and the larger the model is, the larger the MAC is. This causes a contradiction between the network's efficiency and using the regular "group convolution" to avoid the "depth-wise" convolutions. Furthermore, it also can not satisfy to the second principle. Therefore, we will further explore the parameters' properties in the networks to propose novel convolutional kernels meeting the requirements.

2.3. Parameters' properties in the networks

Initially, Fig. 2 plots the distributions of the WideResNets' weights to study their efficiencies. The number of parameters of WideResNet ($k = 8$) is larger than that of WideResNet ($k = 4$). In Fig. 2, the weight distribution of WideResNet ($k = 8$) concentrates on zero area much more heavily compared to WideResNet ($k = 4$). This suggests that the networks with large model size may possess more ineffective parameters (zero parameters). Therefore, the ratio of effective parameters of networks is decreased with the increasing model size. However, in practice, the networks are usually designed by increasing the model size to obtain better performances. This is not an ideal approach to construct networks.

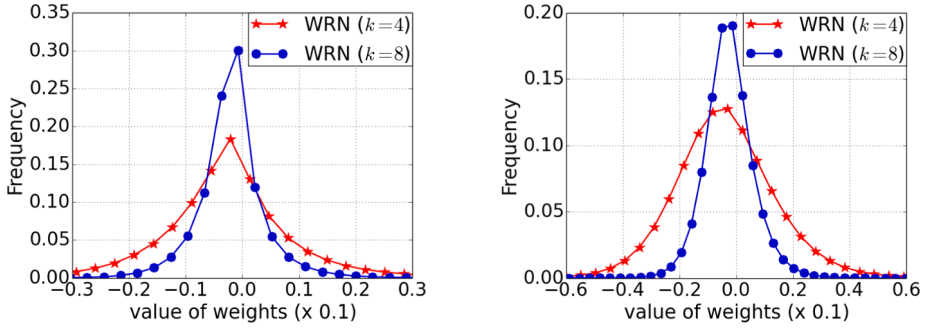
To preserve the effectiveness of the parameters with meeting the performance requirements of the networks, the filters' properties are further explored. Fig. 3 is the statistics of the top-25% maximum activation maps obtained from 64 filters in the first stage of the WideResNets ($k = 4$). In this figure, it is apparent that, in all the 10 categories, the filters with lager contributions almost concentrate on a few same filters, which are marked by the red dotted lines. This implies these filters most probably retrieve the common features of all the classes. For the other filters with smaller contributions, they are also very few and their distributions are more dispersed in all the classes, indicating that these filters are utilized to produce the individualized information for various classes. Finally, these phenomena reflect an assumption that the filters may be classified into two categories: **common** filters and **individualized** filters.

2.4. Motivations

According to the previous design principles and the study of the parameters' properties, since the numbers of common filters and individualized filters are both very small, mobile networks can be directly constructed with only a few filters to reduce the model size and force the filters to learn and generalize, which is very different from the traditional design manner. However, as a common practice, only employing a few filters can not preserve the width of the networks, resulting in the less richness of the obtained features and low performances. Inspired from the traditional regular convolutions, who share their filters in the spatial extent, we want to share the filters more thoroughly to reduce the model size without performance loss. Accordingly, for the common filters, in order to utilize their high contributions, this paper proposes two shared mechanisms to reuse them comprehensively. For the individualized filters, they can only apply the low-cost "point-wise" filters to preserve the small model size. Furthermore, the individualized filters can also utilize one shared mechanism to further decrease the model size, which will be demonstrated in the following section. Based on these designs, this paper proposes Highly Shared Convolutional Networks (HSC-Nets).

Our contributions are listed as bellow:

(1) The main factors of the networks' inefficiency both in the training and test are first explored to determine the design principles of the mobile models. Then, the parameters' properties are studied. Experiments implied the filters may be classified into common filters and individualized filters.
(2) This paper proposes the HSC-Nets, which employ two shared methods. The HSC-Nets can share the filters much more thoroughly than the traditional mobile networks, largely decrease the model size and avoid utilizing the inefficient "depth-wise" convolutions, contributing to significantly reducing the MAC and test runtime.
(3) By using the shared manners, every filter's generalization is improved through processing much more information, leading to no more training epoches addition and more efficiency in training than the other group-conv models.
(4) Experimental results showed that, compared with the other group-conv mobile networks, the HSC-Nets are more efficient in both the training and test processes.



(a) On the CIFAR-10 (b) On the CIFAR-100

Fig. 2. Distributions of the WideResNets' weights.

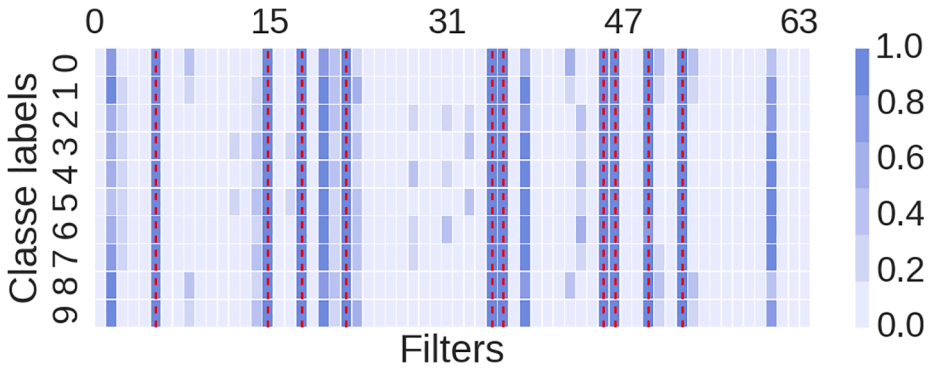


Fig. 3. Distribution of top-25% maximum activation maps obtained by 64 filters from the first stage of the WideResNets ($k = 4$). As the definitions in Zagoruyko and Komodakis (2016), k indicates the expand factor of the basic width. The networks are performed on CIFAR-10 datasets. The final values are computed from the CIFAR-10 test datasets with 1000 images i.n every classes.

3. Highly Shared Convolutional Neural Networks

This section firstly illustrates two shared mechanisms. Then, the detailed structure of the Highly Shared Convolutional Neural Networks (HSC-Nets) is introduced.

3.1. Repeated shared mechanism

According to the motivations in Section 2.4, since the common filters' activation contributions are much higher than the other filters, these filters can be reused multiple times to preserve the retrieved features' richness with timely decreasing the model size. The common filters can be reused form two different levels: channel extent and layer extent. The relevant filter can be called Channel Repeated Filter (CRF) and Layer Repeated Filter (LRF), respectively.

Channel repeated filter. Suppose a 4-D regular kernel is $\mathcal{W}$ ($\mathcal{W} \in \mathbb{R}^{k\times k\times C\times D}$), where k, C and D represent the spatial size, number of input and output channels, respectively. Thus, from Fig. 1(a) and (b), the regular filter's size is ($k \times k \times C \times D$) and the group filter's size is decreased to $\frac{(k\times k\times C\times D)}{G}$, where G is the number of the groups. The Channel Repeated Filter (CRF) is proposed based on the group filter. If the channel repeated times are R, CRF will only utilize $\frac{1}{R}$ filters of the group convolutions' and reuse them R times along the channel extent, which is shown in Fig. 1(c). Additionally, CRF can be seen as sharing the filters in the channel extent with the stride $\frac{C}{G}$. Thus, the CRF kernel's size is $\frac{(k\times k\times C\times D)}{G\times R}$.

Layer Repeated Filter. The Block Term Decomposition (De Lathauwer, 2008) can elegantly illustrate the bottleneck structure with different spatial convolutional layers from the mathematical view. Suppose a basic module has L layers, they can be stacked together along the layer extent and the final filters will be $\mathcal{W}$ ($\mathcal{W} \in \mathbb{R}^{k\times k\times C\times (D\times L)}$). Based on block term decomposition, they can be decomposed as following:

$$\mathcal{W} = \sum_{g=0}^{G-1} \mathcal{S}_g \bullet_3 \mathcal{A}_g^{(3)} \bullet_4 \mathcal{A}_g^{(4)},$$

$$\text{where} \begin{cases} \mathcal{S}_g \in \mathbb{R}^{k\times k\times \frac{C^*}{G}\times \frac{D^*}{G}} \\ \mathcal{A}_g^{(3)} \in \mathbb{R}^{1\times 1\times C\times \frac{C^*}{G}} \\ \mathcal{A}_g^{(4)} \in \mathbb{R}^{1\times 1\times (D\times L)\times \frac{D^*}{G}} \end{cases} \tag{1}$$

In the Eq. (1), the convolutional filters from different layers are decomposed from the input (the third) axis and the output (the fourth) axis, where g indicates the g^{th} group. $\bullet_3$ and $\bullet_4$ indicate the *mode* -3 and the *mode* -4 product, respectively. C^* and D^* denote the low dimensional space corresponding to C and D, respectively. $\mathcal{S}_g$ is the g^{th} decomposed core term. And $\mathcal{A}_g^{(3)}$ and $\mathcal{A}_g^{(4)}$ represent the g^{th} obtained tensors from the *mode* -3 and the *mode* -4, respectively. From this equation, $\mathcal{A}_r^{(3)}$ and $\mathcal{A}_r^{(4)}$ are both the "point-wise" convolutions, which can illustrate the bottleneck structure. Furthermore, in all the L layers, $\mathcal{A}_r^{(3)}$ and $\mathcal{S}_r$ are the same, and only the $\mathcal{A}_r^{(4)}$ is different. Consequently, the filters are further shared at different layers in this paper, which is called Layer Repeated Filter (LRC). Further, if $\mathcal{S}_i = \mathcal{S}_j (i \neq j, i, j \in \{0, 1, 2, \ldots, G-1\})$ in Eq. (1), it implies the filters can be shared from both the channel and layer extents.

3.2. Arranged shared mechanism

Suppose a 3-D filter $\mathcal{W}$ ($\mathcal{W} \in \mathbb{R}^{k\times k\times C}$, C and k denote the number of input channels and spatial size, respectively) and the input feature maps $\mathcal{I}$ ($\mathcal{I} \in \mathbb{R}^{s\times s\times C}$, s is the input spatial size), the output $\mathcal{O} = \sum_{i=0}^{C-1} \mathcal{W}_i \otimes \mathcal{I}_i$, where $\otimes$ indicates the convolutional operation. Therefore, the final output is the summation of the multiplications between each plane filter and the corresponding input feature map. In accordance with the discovery in Section 2.3, the common filters possess much higher contributions than the other filters. This suggests that, for a filter, its better representations are not only obtained from the entire the filter but also most probably from each plane filter. Based on this analysis, the plane filters in one 3-D filter can be reused multiple times by re-arranging them along the channel extents, leading to constructing new filters without increasing the number of parameters. This filter (see Fig. 4) is called Arranged Shared Filter (ASF) and denoted by $\mathcal{W}_T^A$, where T indicates the shared times. Therefore, a 4-D $\mathcal{W}_T^A$ can be obtained as below:

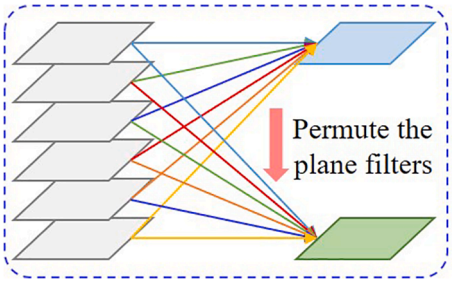


Fig. 4. ASF filter. Different colored lines indicate various plane filters.

$$\mathcal{W}_T^A = Concat_{t=0}^{T-1}(P_t\mathcal{W}_b), \text{ where } \begin{cases} \mathcal{W}_b \in \mathbb{R}^{k\times k\times C\times \frac{D}{T}} \\ \mathcal{W}_T^A \in \mathbb{R}^{k\times k\times C\times D} \end{cases} \tag{2}$$

where D is the output channels. *Concat* is the 0catenation operation of the filters along the output extent. $\mathcal{W}_b$ denotes the basic filter to be shared, and P_t represents the t^{th} permutation of the plane filters in each 3-D filter along the input axis. Therefore, according to the Eq. (2), the filter size reduces T times. Finally, the arranged shared mechanism can be also applied to the basis of the repeated shared manner. The filter's structure shared by the both two methods is shown in Fig. 1(d). From this figure, apparently, the proposed shared methods can reuse the filters from all the extents and more completely than the traditional regular convolutions.

Algorithm 1. : The algorithm of basic module of HSC-Nets. The superscripts of L, C and A denote the **Layer**, **Channel** and **Arranged** shared manners, respectively. $k \times k$ indicates the spatial size of filters. $\boldsymbol{\theta}$ represents the parameters of the relevant layers.

1: Initialize:
The transformation of the first layer from bottleneck $f_{1\times1}^{L,A}(\cdot;\boldsymbol{\theta}_1)$;
The transformation of the last layer from bottleneck $f_{1\times1}^{A}(\cdot;\boldsymbol{\theta}_2)$ and $f_{1\times1}^{A}(\cdot;\boldsymbol{\theta}_3)$;
The transformation of the group spatial filter layer between the bottleneck $f_{k\times k}^{C,L,A}(\cdot;\boldsymbol{\theta}_4)$;
The transformation of layer shared group point-wise filter layer: $f_{1\times1}^{L}(\cdot;\boldsymbol{\theta}_5)$;
The number of input and output channels of module: M;
The number of channels in bottleneck: K.
The repeated times of arranged shared filters $f^A : T$;
The repeated times of channel shared filters f^C at the channel extent: R.
The number of groups of group filters: G.
2: **Repeat:**
3: Module input: $\boldsymbol{x}$
4: First stage: $\boldsymbol{y} = f_{1\times1}^{L,A}(\boldsymbol{x};\boldsymbol{\theta}_1,K,T)$
5: $\boldsymbol{y_1} = f_{k\times k}^{C,L,A}(\boldsymbol{y};\boldsymbol{\theta}_4,K,G,R,T)$
6: $\boldsymbol{y_2} = f_{1\times1}^{L}(\boldsymbol{y};\boldsymbol{\theta}_5,K,G)$
7: $\boldsymbol{y} = \boldsymbol{y_1} + \boldsymbol{y_2}$
8: $\boldsymbol{y} = f_{1\times1}^{A}(\boldsymbol{y};\boldsymbol{\theta}_2,M,T)$
9: Second stage: $\boldsymbol{y} = f_{1\times1}^{L,A}(\boldsymbol{y};\boldsymbol{\theta}_1,K,T)$
10: $\boldsymbol{y_1} = f_{k\times k}^{C,L,A}(\boldsymbol{y};\boldsymbol{\theta}_4,K,G,R,T)$
11: $\boldsymbol{y_2} = f_{1\times1}^{L}(\boldsymbol{y};\boldsymbol{\theta}_5,K,G)$
12: $\boldsymbol{y} = \boldsymbol{y_1} + \boldsymbol{y_2}$
13: $\boldsymbol{y} = f_{1\times1}^{A}(\boldsymbol{y};\boldsymbol{\theta}_3,M,T)$
14: Compute loss and update $\boldsymbol{\theta}$
15: **Until convergence**

3.3. The structure of the HSC-Nets

According to the analysis in Section 2.3 and Eq. (1), the Highly Shared Convolutional Neural Networks (HSC-Nets) are proposed. Since the structure of the WideResNet (Zagoruyko & Komodakis, 2016) is simple and it can perform satisfactorily on many challenging datasets, our HSC-Nets employ its global structure. The HSC-Net also contains three big blocks (stages) with different output spatial sizes (*e.g.*, "32×32", "16×16" and "8×8") and every block has B basic modules. The output channels of every stage will be double when stepping into the next stage.

From the previous illustrations of the channel, layer and arranged shared manners, apparently, they are orthogonal and can be combined with each other in the HSC-Nets. The basic HSC-Net's module is shown in the Fig. 5. Similar to WideResNet, it also includes two general layers. However, in the HSC module, these two general layers are decomposed as the Eq. (1). The filters represented by the red and orange circles apply at least two shared methods, which are designed inspired by the common filters. To keep the features' richness, the common filters are supplemented with the individualized filters, which are the "group point-wise" convolutions marked by green circles. The filters colored blue are the last bottleneck in each general layer, they can also use the arranged shared method to further decrease the model size. Thus, the HSC-Nets share the filters much more completely than the other models. The detailed structure of basic HSC module is demonstrated in Algorithm 1.

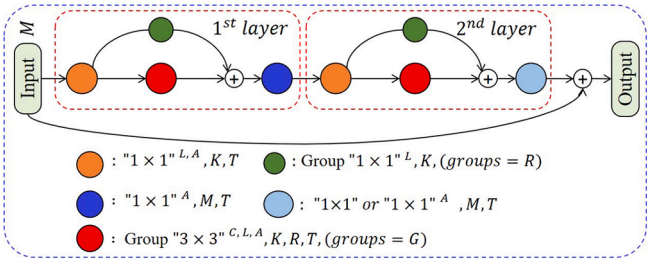


Fig. 5. The HSC-Net's basic module. M, K indicate the input channels and the bottleneck's output channels, respectively. T is the repeated times of arranged shared filters. The superscripts of L, C and A denote the **Layer**, **Channel** and **Arranged** shared manners, respectively.

4. Experimental results and analysis

The HSC filter and the entire structure of the HSC-Nets are proposed and implemented based on the "group convolutions" and WideResNets, respectively. Therefore, the HSC-Nets are mainly compared with the group-conv mobile networks and WideResNets.

4.1. Datasets

Low resolution ImageNet datasets (Chrabaszcz, Loshchilov, & Hutter, 2017) are the down-sampled variants of **regular ImageNet** (Deng et al., 2009) and include the same number of classes and images of the regular ImageNet. HSC-Nets are performed on the ImageNet-32×32. The augmentations of the datasets are the same with (Chrabaszcz et al., 2017). As demonstrated in Chrabaszcz et al. (2017), since the low resolution ImageNet databases have much less spatial information, leading to much more difficulty in these databases compared to the original regular ImageNet datasets.

Tiny ImageNet[7] is a subset of regular ImageNet (Deng et al., 2009). It has 200 classes, which are sampled from 1000 categories in the regular ImageNet. Every class contains 500 training images, 50 validation images and 50 testing images. All the spatial sizes of the images are resized to "64×64".

CIFAR datasets (Krizhevsky & Hinton, 2009) include CIFAR-10 and CIFAR-100 with 10 and 100 classes, respectively. They both have $60,000$ colored nature scene images in total and the images' size is "32×32". These two datasets both contain $50,000$ images for training and $10,000$ images for testing.

CINIC-10 dataset (Darlow, Crowley, Antoniou, & Storkey, 2018) is an extension of CIFAR-10 via the addition of down-sampled ImageNet images. It is composed of $270,000$ colored images with spatial size "32×32" in 10 classes. These images are split into three equal-sized train, validation, and test subsets.

Following the common practices (He et al., 2016; Huang, Sun, Liu, Sedra, & Weinberger, 2016), the same data augmentations are applied to these databases and the same learning initializations on these datasets are also utilized. The training epoches are set to 40 on the low resolution ImageNet and 200 on the Tiny ImageNet, CINIC-10 and CIFAR databases. The SGD method and Nesterov momentum (Sutskever, Martens, Dahl, & Hinton, 2013) are employed to optimize. On the low resolution ImageNet, the learning rate starts from 0.01 and is divided by 5 at the 10^{th}, 20^{th} and 30^{th} epoch. On the tiny ImageNet and CINIC-10, the learning rate starts from 0.1 and is divided by 10 at the 100^{th}, 150^{th} and

[7] https://tiny-imagenet.herokuapp.com/.

175^{th} epoch. On the CIFAR, the learning rate starts from 0.1 and is divided by 5 at the 60^{th}, 120^{th} and 160^{th} epoch. The momentum is 0.9, and the weight decay is set to $5e-4$ on low resolution ImageNet and CIFAR-100, $1e-4$ on the tiny ImageNet and CINIC-10, and $2e-4$ on the CIFAR-10. Finally, the mini-batch size is set to 128.

4.2. Initializations of the HSC-Nets

Different versions of the HSC-Nets and WideResNets are respectively denoted by HSC-Net-*M*-*G*-*R*-*B*-α and WRN-*B*-*k*, where *G* and *R* indicate the groups and repeated times at the channel extent, respectively. *B* is the modules' number in a big block. *M* denotes the width in the first block. And α is the divided factor of the output channels in bottleneck. Finally, *k* represents the expansion factor of every block's width in WRN (see Zagoruyko & Komodakis, 2016). These models' sizes can be changed by toggling $M,\ G,\ R,\ \alpha$ or *k*.

Since automatic parameter tuning system will need additional modules and operations in CNNs, this will lead to more complex structure of networks and time-consuming training. Furthermore, it is also impossible to search all possible cases of hyper-parameters due to the tremendous amount of training time. Therefore, following the common practices (MobileNet family Howard et al., 2017; Sandler et al., 2018, IGCV family Zhang et al., 2017; Xie et al., 2018; Sun et al., 2018 and ShuffleNet family Zhang et al., 2018; Ma et al., 2018 networks), we also utilize the rough searches to determine the final hyper-parameters. Furthermore, since the large number of the groups in convolutions will increase the runtime, in order to preserve a better tradeoff between the runtime and groups, *G* is set to 16 and 8, which is determined by our large number of the evaluations in practice, togging *G* in HSC-Nets and testing them. By the same way, the repeated times *R* can be set to 16, 8 and 4 to decrease the model size. Finally, for the arranged shared method, in order to implement the arranged shared filters efficiently and disrupt the original order of the plane filters as much as possible, the "shuffle" operation is used to permute the plane filters in every 3-D filter. This "shuffle" operation is similar to that in the ShuffleNet family networks (Zhang et al., 2018; Ma et al., 2018). Since the arranged shared filters are largely employed in the individualized filters and these filters should have distinguished representations with high-quality, the repeated times *T* is only set to 2, which is enough to meet the requirements of reducing model size and preserving the richness of retrieved features.

4.3. Experiments on the low resolution ImageNet

Initially, since the entire structure of HSC-Nets are implemented based on the WideResNet, the HSC-Nets are compared with the WideResNets under approximately the same performance. Table 1 displays the model size, accuracy and reduction rate, respectively. Evidently, the HSC-Nets can significantly reduce the model size and even perform better than some WRNs. In particular, HSC-Net-*M*224-*G*16-*R*8-α1.75 produces competitive Top-1 and Top-5 accuracies with **5.3X** fewer parameters. Moreover, it also achieves satisfactory Top-5 accuracy compared with the Top-1 accuracy obtained by the state-of-the-art models on regular ImageNet.

Additionally, since the HSC filters are proposed based on the "group convolutions", the HSC-Nets are compared with the other latest group-conv mobile models. Because the number of Flops can not evaluate the efficiency of the mobile networks properly in practice (see Section 2.1), the training time on GPU and test inference runtime on CPU are evaluated and depicted in the Table 2. It is apparent that, compared with the WRN ($k = 3,\ B = 4$) (see Table 1, $params: 3.5M$, $Top-1\ acc: 48.94\%$), MobileNetV2 and IGCV3 obtain only a little accuracy improvement. Although the ShuffleNetV2 is the latest state-of-the-art mobile networks, it achieves even worse accuracy than the regular network WRN ($k = 3,\ B = 4$). However, our HSC-Net-*M*128-*G*16-*R*8-α2 can perform much better than those networks with utilizing only 2.3*M* parameters. Furthermore, its training and inference time is also effectively decreased among all the group-conv mobile networks. Especially, under approximately the same accuracy, the training cost, inference runtime and model size of the HSC-Net-*M*96-*G*8-*R*4-α2 are all the least compared with the other models. Therefore, this can prove our network is more efficient both in the training and test processes.

4.4. Experiments on tiny ImageNet and CINIC

As illustrated in the literature (Darlow et al., 2018), since the low resolution ImageNet is very difficult, CINIC-10 and tiny ImageNet datasets are constructed. Especially, CINIC-10 possesses a fair assessment of generalization performance. Table 3 proves that HSC-Net can effectively improve the performance. On tiny ImageNet, the accuracy of HSC-Nets is increased by 5.87% compared with the IGCV3. Also, on CINIC-10, HSC-Nets perform the best among all the compared networks. Therefore, the experiments verify the effectiveness of the HSC-Nets. It also proves that the HSC-Net generalizes better than the other models, because the shared mechanisms can improve the parameters' generalization by processing much more information in a filter and obtaining the gradients from different channels and layers.

4.5. Experiments on CIFAR

On CIFAR datasets, the HSC-Nets are also compared to other state-of-the-art group-conv mobile models. In Table 4, it is evident that the HSC-Nets achieve the best performances with much smaller model size. Moreover, in the training process, our HSC-Nets are also the most efficient with only 200 training epoches. However, the epoch numbers of the MobileNetV2, IGCV family and ShuffleNetV2 should be very large (with 400 training epoches) to force the "depth-wise" filters used in these models to generalize well. Consequently, according to the evaluations from all the aspects, the HSC-Net is much more efficient than the other group-conv mobile models both in the training and test processes.

Table 1
Accuracy (%) comparisons between HSC-Nets and WRNs on ImageNet-32 × 32. In all the HSC-Nets, $B = 4$. The accuracies are the average results of 5 runs.

Method	Params	Top-1	Top-5	Reduction
WRN ($k = 3$) (our imple.)	3.5 *M*	48.94%	73.92%	**2.5X**
HSC-Net-*M*96-*G*8-*R*4-α2	1.4 *M*	49.32%	74.50%	
WRN ($k = 5$) Chrabaszcz et al. (2017)	9.5 *M*	54.54%	78.74%	**2.7X**
HSC-Net-*M*160-*G*8-*R*4-α2	3.5 *M*	55.76%	79.25%	
WRN ($k = 10$) Chrabaszcz et al. (2017)	37.1 *M*	59.04%	81.13%	**5.3X**
HSC-Net-*M*224-*G*16-*R*8-α1.75	7.0 *M*	58.98%	81.69%	

Table 2
Comparisons of accuracy (%) on ImageNet-32 × 32. CPU^{test} indicates the test runtime on the Intel ® Xeon(R) CPU E5-2620 v4 processor. GPU^{train} denotes the runtime of training a model on 4 TITAN Xp GPUs. The accuracies and runtime are the average results of 5 runs.

Method	Params	Top-1 acc.	GPU^{train}	CPU^{test}
MobileNetV2 1.0 ×	3.5*M*	48.98%	142.2 *h*	151 *ms*
IGCV3-D 1.0 ×	3.5*M*	49.40%	257.5 *h*	295 *ms*
ShuffleNetV2 2.0 ×	7.5*M*	48.46%	124.7 *h*	135 *ms*
HSC-Net-*M*96-*G*8-*R*4-α2	**1.4M**	49.32%	**79.4 h**	**82 ms**
HSC-Net-*M*128-*G*16-*R*8-α2	2.3*M*	**53.41%**	96.7 *h*	99 *ms*

Table 3
Comparisons of accuracy (%) with the latest state-of-the-art mobile models on tiny ImageNet and CINIC-10. The accuracies are the average results of 5 runs.

Method	Params	tiny ImageNet acc.	CINIC-10 acc.
ResNet-18 (pre-act) Darlow et al. (2018)	11.2*M*	–	87.16%
MobileNetV1 Darlow et al. (2018)	3.2*M*	–	80.45%
MobileNetV2 (1.25 ×)	3.8*M*	59.55%	85.43%
IGCV3-D (1.25 ×)	3.7*M*	59.94%	85.35%
ShuffleNetV2 (2.0 ×)	5.5*M*	59.17%	83.94%
HSC-Net-*M*128-*G*16-*R*4-*α*2	**1.8M**	**65.81%**	**87.99%**

Table 4
Accuracy (%) comparisons to other state-of-the-art mobile architectures on CIFAR. ★ denotes the arranged shared mechanism is also utilized at the last "point-wise" layer in every module (see Fig. 5). The accuracies are the average results of 5 runs.

Method	Params	CIFAR-10 acc.	CIFAR-100 acc.
MobileNetV2 (reported in Sun et al., 2018)	2.3*M*	94.56%	77.09%
IGCV2 (reported in Sun et al., 2018)	2.3*M*	94.76%	77.45%
IGCV3-D 1.0× Sun et al. (2018)	2.4*M*	94.96%	77.95%
ShuffleNetV2 2.0× (our imple.)	5.5*M*	93.77%	75.17%
HSC-Net-*M*128-*G*16-*R*8-*α*2	**1.8M**	**95.92%**	**78.28%**
HSC-Net-*M*192-*G*16-*R*8-*α*2 ★	3.2*M*	**96.05%**	**80.01%**

4.6. Experiments on regular ImageNet

In the HSC-Nets, the filters are shared much more comprehensively compared with the other networks, which results in the lack of richness in the HSC filters. Accordingly, with the increasing of the information richness on the regular resolution datasets, the HSC-Nets' representational ability are not sufficient enough. However, HSC-Nets are still evaluated and compared to other group-conv mobile networks on regular ImageNet. In Table 5, the HSC-Net is composed of 4 big blocks and these blocks include 3, 3, 2 and 2 basic modules, respectively. The output channels in every stage are 160, 320, 480, and 1120. The experimental results show that, compared to these mobile networks, our HSC-Net can still produce a competitive performance. Moreover, the test runtime and training epoches of the HSC-Nets are both the least (training epoches: HSC-Net: 120 epoches, MobileNetV2 & IGCV3: 480 epoches, ShuffleNetV2: 250 epoches). Consequently, HSC-Net is also a promising network for the regular resolution datasets. In the future, it is a valuable study issue to propose shared filters with more abundant patterns and larger shared degrees to drive the shared networks to perform better.

Table 5
Comparisons on regular ImageNet datasets. CPU^{test} indicates the test runtime on the Intel ® Xeon(R) CPU E5-2620 v4 processor. $Epoch^{train}$ denotes the number of training epoches in different networks. The accuracies and runtime are the average results of 5 runs.

Method	Params	Top-1 acc.	CPU^{test}	$Epoch^{train}$
ResNet-101 (1 × 64*d*) He et al. (2016)	44.5 *M*	78.0%	400 *ms*	120
MobileNetV2 1.4 × Sandler et al. (2018)	6.9 *M*	74.7%	272 *ms*	480
IGCV3 1.4× Sun et al. (2018)	7.2 *M*	74.5%	505 *ms*	480
SE-ShuffleNetv2 2.0× Ma et al. (2018)	8.8 *M*	75.4%	235 *ms*	250
HSC-Nets-*M*160 (ours)	7.2 *M*	74.2%	**189 ms**	120

4.7. Ablation study of the shared mechanisms

In order to explore the effectiveness of the shared mechanism, the l_1 norms of weights from the HSC-Nets and the WideResNets are computed respectively. The width and the number of modules of these two networks are the same. The final frequency distributions of the l_1 norms are shown in the Fig. 6. It is clear that, on both two CIFAR datasets, the l_1 norms of the WideResNets mostly distribute near the zero. However, the l_1 norms of the HSC-Nets are more dispersed than that of the WideResNets. Accordingly, compared with the unshared WideResNets, the parameters' efficiency and representational abilities of the shared HSC-Nets have been effectively improved. This is because the shared mechanisms force every filter to process much more information, contributing to improving the generalization and no more addition of the training epoches. Therefore, the HSC-Nets can perform better and be trained more efficiently than the other mobile group-conv models.

Following this study, the high contribution maps activated by over 90% and 80% sampled images in CIFAR-10 test datasets have been counted in Fig. 7(a) and (b), respectively. The comparisons are also between the HSC-Net-*M*128-*G*16-*R*8-*α*2 and the WideResNets ($k = 8$) with the same width and number of modules. The accuracies achieved by HSC-Nets and WideResNets are 95.92% and 95.80% with 1.8*M* and 23.4*M* parameters, respectively. Notably, the number of the high contribution maps of the HSC-Nets is much larger than that of the WideResNets. This further proves the high efficiency and effectiveness of the proposed shared mechanisms. It also illustrates why the HSC-Nets perform better with much smaller model size.

To further explore the properties of the shared mechanisms, we also compute the HFS-Net's top-48 maximum activation maps of every classes. The samples are all the CIFAR-10 test databases. And, the activation maps are from the first stage's outputs, which are obtained by 128 filters in HFS-Net-*M*128-*G*16-*R*8★. The final distributions are shown in the Fig. 8. From this figure, although the arranged shared filters displayed at the right side of the orange line are generated based on the filters from the left side, the high contribution filters (marked by red dotted lines) distribute at the both sides of the orange line. This proves that the arranged filters are also very effective in the HSC-Nets.

Finally, the top-3 maximum high contribution original and shared filters are visualized by non-parameterization manner in the Fig. 9, respectively. This experiment is to verify if these two kinds of filters are different. Apparently, the patterns of the original filters (in the first row) and shared filters (in the second row) can be easily distinguished. Therefore, although the shared filters are obtained from the original filters, they can also possess high contributions and retrieve different geometric characteristics, contributing to preserving the richness of the retrieved features.

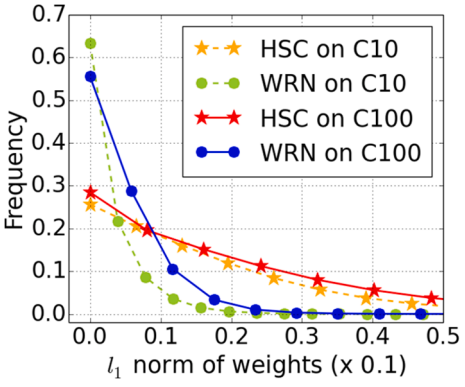
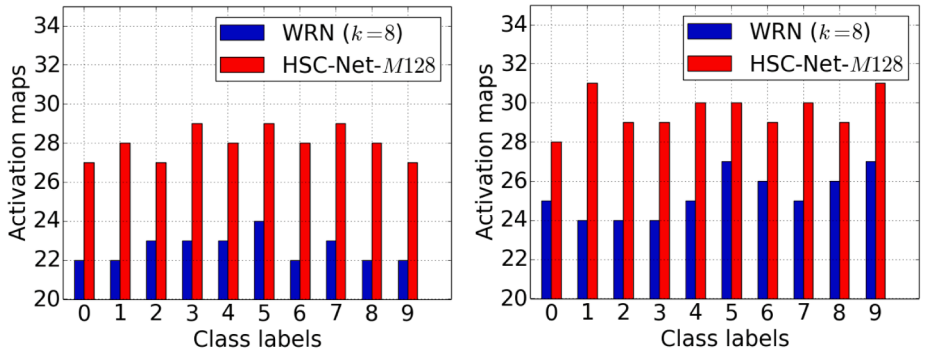


Fig. 6. l_1 norm distributions on CIFAR.



(a) Over 90% samples activated (b) Over 80% samples activated

Fig. 7. Comparisons of the high contribution maps activated by over 90% and 80% sampled images in CIFAR-10.

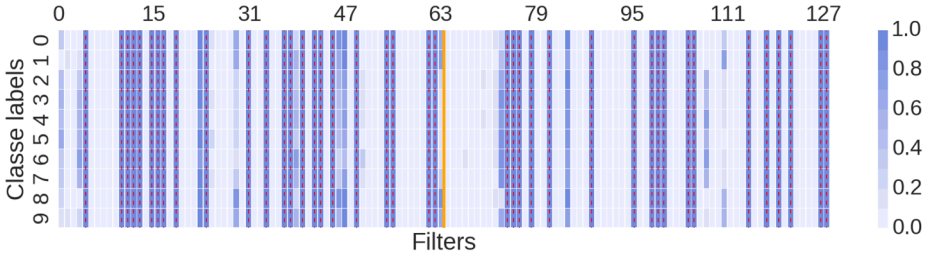
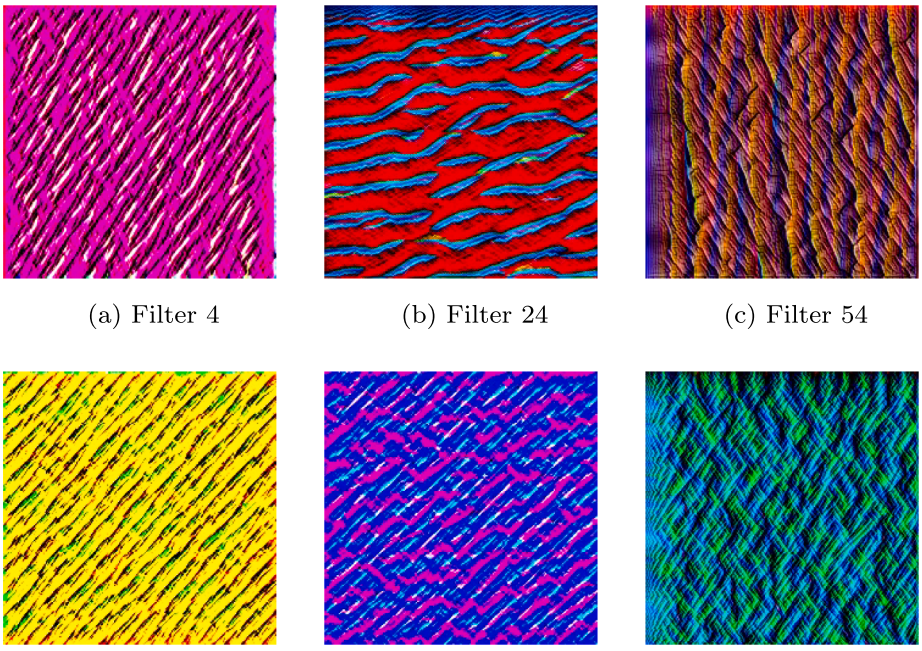


Fig. 8. Distribution of top-48 maximum activation maps obtained by 128 filters from the first stage of the HFS-Net-*M*128-*G*16-*R*8★. The filters displayed at the right of the orange line are obtained by the arranged shared method.



(a) Filter 4 (b) Filter 24 (c) Filter 54

(d) Filter 75 (e) Filter 81 (f) Filter 104

Fig. 9. Visualizations of the top-3 maximum high contribution original and shared filters of the HSC-Net's first stage. The model is trained on the CIFAR-10. The filters in the second row utilize the arranged shared mechanism.

5. Conclusions

To propose more efficient mobile networks both in the training and test, this paper firstly found that the "depth-wise" convolution is the main causes of the network's inefficiency. Then, we discovered the filters with high contributions are very few and these filters can be classified into common and individualized filters. Inspired from these explorations and following the group-conv models, the HSC-Nets are directly designed with very few filters through two shared manners. Compared with the other mobile networks, the HSC-Nets can share the filters more completely, leading to avoiding the "depth-wise" convolutions. Also, the shared mechanisms can improve the filter's

generalization significantly, resulting in no more training epoches addition. Experiments verified that, compared with the latest group-conv mobile CNNs, HSC-Nets effectively decrease both the training and test runtime.

CRediT authorship contribution statement

Yao Lu: Conceptualization, Methodology, Writing - original draft. **Guangming Lu:** Writing - review & editing, Supervision, Project administration, Funding acquisition. **Yicong Zhou:** Validation, Writing - review & editing, Formal analysis. **Jinxing Li:** Validation, Writing - review & editing, Formal analysis. **Yuanrong Xu:** Visualization, Resources. **David Zhang:** Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported in part by the Open Project Fund (AC01202005018, AC01202005017) from Shenzhen Institute of Artificial Intelligence and Robotics for Society, in part by National Key Research and Development Program of China under Project Number 2018AAA0100102, in part by the Shenzhen Fundamental Research Fund under Grant JCYJ20180306172023949, in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2019B1515120055, in part by NSFC fund under Grant 61906162, in part by the China Postdoctoral Science Foundation (2019TQ0316, 2019M662198), in part by the Medical Biometrics Perception and Analysis Engineering Laboratory, Shenzhen, China, in part by the Science and Technology Development Fund, Macau SAR (File no. 189/2017/A3), and in part by University of Macau (File no. MYRG2018-00136-FST).

References

Chrabaszcz, P., Loshchilov, I., & Hutter, F. (2017). A downsampled variant of imagenet as an alternative to the CIFAR datasets. CoRR, abs/1707.08819. URL: http://arxiv.org/abs/1707.08819. arXiv:1707.08819.

Darlow, L. N., Crowley, E. J., Antoniou, A., & Storkey, A. J. (2018). CINIC-10 is not imagenet or CIFAR-10. CoRR, abs/1810.03505. URL: http://arxiv.org/abs/1810.03505. arXiv:1810.03505.

De Lathauwer, L. (2008). Decompositions of a higher-order tensor in block terms-part ii: Definitions and uniqueness. *SIAM Journal on Matrix Analysis and Applications, 30*, 1033–1066. https://doi.org/10.1137/060661685. URL: https://doi.org/10.1137/070690729.

Deng, J., Dong, W., Socher, R., Li, L., Li, Kai, & Fei-Fei, Li (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248–255). https://doi.org/10.1109/CVPR.2009.5206848. URL: https://doi.org/10.1109/CVPR.2009.5206848.

Dong, X., Chen, S., & Pan, S. (2017). Learning to prune deep neural networks via layer-wise optimal brain surgeon. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems 30* (pp. 4857–4867). Curran Associates, Inc. URL: http://papers.nips.cc/paper/7071-learning-to-prune-deep-neural-networks-via-layer-wise-optimal-brain-surgeon.pdf.

Gao, H., Wang, Z., Cai, L., & Ji, S. (2020). Channelnets: Compact and efficient convolutional neural networks via channel-wise convolutions. In *IEEE Transactions on Pattern Analysis and Machine Intelligence* (p. 1). https://doi.org/10.1109/TPAMI.2020.2975796

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 770–778). https://doi.org/10.1109/CVPR.2016.90. URL: https://doi.org/10.1109%2FCVPR.2016.90.

He, Y., Zhang, X., & Sun, J. (2017). Channel pruning for accelerating very deep neural networks. In *2017 IEEE international conference on computer vision (ICCV)* (pp. 1398–1406). https://doi.org/10.1109/ICCV.2017.155. URL: https://www.computer.org/10.1109/ICCV.2017.155.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. CoRR, abs/1704.04861. URL:http://arxiv.org/abs/1704.04861. arXiv:1704.04861.

Huang, G., Liu, S., Maaten, L. v. d., & Weinberger, K. Q. (2018). Condensenet: An efficient densenet using learned group convolutions. In *2018 IEEE/CVF conference on computer vision and pattern recognition* (pp. 2752–2761). doi: 10.1109/CVPR.2018.00291.

Huang, G., Sun, Y., Liu, Z., Sedra, D., & Weinberger, K. (2016). Deep networks with stochastic depth. In *Computer vision – ECCV 2016* (Vol. 9908, pp. 646–661). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-46493-0_39

Hu, Y., Sun, S., Li, J., Wang, X., & Gu, Q. (2018). A novel channel pruning method for deep neural network compression. CoRR, abs/1805.11394. URL:http://arxiv.org/abs/1805.11394. arXiv:1805.11394.

Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images. Technical Report Citeseer. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.222.9220&rep=rep1&type=pdf.

Lauret, P., Fock, E., & Mara, T. A. (2006). A node pruning algorithm based on a fourier amplitude sensitivity test method. *IEEE Transactions on Neural Networks, 17*, 273–293. https://doi.org/10.1109/TNN.2006.871707. URL: https://doi.org/10.1109/TNN.2006.871707.

Lin, J., Rao, Y., Lu, J., & Zhou, J. (2017). Runtime neural pruning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems 30* (pp. 2181–2191). Curran Associates, Inc. URL: http://papers.nips.cc/paper/6813-runtime-neural-pruning.pdf.

Liu, C., Zoph, B., Neumann, M., Shlens, J., Hua, W., Li, L.-J., Fei-Fei, L., Yuille, A., Huang, J., & Murphy, K. (2018). Progressive neural architecture search. In V. Ferrari, M. Hebert, C. Sminchisescu, & Y. Weiss (Eds.), *Computer vision – ECCV 2018* (Vol. 11205, pp. 19–35). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-01246-5_2.

Luo, J. H., Wu, J., & Lin, W. (2017). Thinet: A filter level pruning method for deep neural network compression. In *2017 IEEE international conference on computer vision s (ICCV)* (pp. 5068–5076). volume 1. URL:https://www.computer.org/10.1109/ICCV.2017.541. doi: 10.1109/ICCV.2017.541.

Ma, N., Zhang, X., Zheng, H.-T., & Sun, J. (2018). Shufflenet v2: Practical guidelines for efficient cnn architecture design. In V. Ferrari, M. Hebert, C. Sminchisescu, & Y. Weiss (Eds.), *Computer vision – ECCV 2018* (Vol. 11218, pp. 122–138). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-01264-9_8.

Real, E., Aggarwal, A., Huang, Y., & Le, Q. V. (2018). Regularized evolution for image classifier architecture search. CoRR, abs/1802.01548. URL: http://arxiv.org/abs/1802.01548. arXiv:1802.01548.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. (2018). Mobilenetv 2: Inverted residuals and linear bottlenecks. In *2018 IEEE/CVF conference on computer vision and pattern recognition* (pp. 4510–4520). https://doi.org/10.1109/CVPR.2018.00474

Sun, K., Li, M., Liu, D., & Wang, J. (2018). Igcv 3: Interleaved low-rank group convolutions for efficient deep neural networks. CoRR, abs/1806.00178. URL: https://arxiv.org/abs/1806.00178. arXiv:1804.06202.

Sutskever, I., Martens, J., Dahl, G., & Hinton, G. (2013). On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th international conference on machine learning – Vol. 28 ICML'13* (pp. III-1139–III-1147). JMLR.org. URL: http://dl.acm.org/citation.cfm?id=3042817.3043064.

Wang, J., Xu, C., Yang, X., & Zurada, J. M. (2018). A novel pruning algorithm for smoothing feedforward neural networks based on group lasso method. *IEEE Transactions on Neural Networks and Learning Systems, 29*, 2012–2024. https://doi.org/10.1109/TNNLS.2017.2748585. URL: http://ieeexplore.ieee.org/document/8051266.

Xie, G., Wang, J., Zhang, T., Lai, J., Hong, R., & Qi, G. (2018). Interleaved structured sparse convolutional neural networks. In *2018 IEEE/CVF conference on computer vision and pattern recognition* (pp. 8847–8856). https://doi.org/10.1109/CVPR.2018.00922

Yang, T., Chen, Y., & Sze, V. (2018). Designing energy-efficient convolutional neural networks using energy-aware pruning. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 6071–6079). Vol. 00. URL: doi.ieeecomputersociety.org/10.1109/CVPR.2017.643. doi: 10.1109/CVPR.2017.643.

Zagoruyko, S., & Komodakis, N. (2016). Wide residual networks. In E. R. H. Richard C. Wilson, & W. A. P. Smith (Eds.), *Proceedings of the british machine vision conference (BMVC)* (pp. 87.1–87.12). BMVA Press. URL:https://dx.doi.org/10.5244/C.30.87. doi: 10.5244/C.30.87.

Zhang, T., Qi, G., Xiao, B., & Wang, J. (2017). Interleaved group convolutions. In *2017 IEEE international conference on computer vision (ICCV)* (pp. 4383–4392). https://doi.org/10.1109/ICCV.2017.469

Zhang, X., Zhou, X., Lin, M., & Sun, J. (2018). Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *2018 IEEE/CVF conference on computer vision and pattern recognition* (pp. 6848–6856). https://doi.org/10.1109/CVPR.2018.00716

Zoph, B., Vasudevan, V., Shlens, J., & Le, Q. V. (2018). Learning transferable architectures for scalable image recognition. In *2018 IEEE/CVF conference on computer vision and pattern recognition* (pp. 8697–8710). https://doi.org/10.1109/CVPR.2018.00907