# Super Sparse Convolutional Neural Networks

**Yao Lu,[1] Guangming Lu,[*1] Bob Zhang,[2] Yuanrong Xu,[1] Jinxing Li[3]**
[1]Department of Computer Science and Technology, Harbin Institute of Technology (Shenzhen), Shenzhen, China
[2]Department of Computer and Information Science,University of Macau, Macau
[3]Department of Computing, Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong, China
yaolu_1992@126.com, luguangm@hit.edu.cn, bobzhang@umac.mo, xuyuanrong1988@126.com, csjxli@comp.polyu.edu.hk

## Abstract

To construct small mobile networks without performance loss and address the over-fitting issues caused by the less abundant training datasets, this paper proposes a novel super sparse convolutional (SSC) kernel, and its corresponding network is called SSC-Net. In a SSC kernel, every spatial kernel has only one non-zero parameter and these non-zero spatial positions are all different. The SSC kernel can effectively select the pixels from the feature maps according to its non-zero positions and perform on them. Therefore, SSC can preserve the general characteristics of the geometric and the channels' differences, resulting in preserving the quality of the retrieved features and meeting the general accuracy requirements. Furthermore, SSC can be entirely implemented by the "shift" and "group point-wise" convolutional operations without any spatial kernels (*e.g.*, "$3 \times 3$"). Therefore, SSC is the first method to remove the parameters' redundancy from the both spatial extent and the channel extent, leading to largely decreasing the parameters and Flops as well as further reducing the img2col and col2img operations implemented by the low leveled libraries. Meanwhile, SSC-Net can improve the sparsity and overcome the over-fitting more effectively than the other mobile networks. Comparative experiments were performed on the less abundant CIFAR and low resolution ImageNet datasets. The results showed that the SSC-Nets can significantly decrease the parameters and the computational Flops without any performance losses. Additionally, it can also improve the ability of addressing the over-fitting problem on the more challenging less abundant datasets.

## Introduction and related works

The models' size of Convolutional Neural Networks (CNNs) is usually too large to be deployed on the mobile devices and they often suffer from the over-fitting problem caused by the less abundant datasets. As illustrated in (Wu et al. 2018), most of the learned parameters are close to zero and the activated feature maps from different channels in a layer share similar geometric characteristics. This implies the parameters of the convolutional kernel are very redundant. Therefore, numerous methods, which can be classified into two categories, have been proposed to compress the networks. The algorithms in the first category are mostly based on pruning the weights or neurons and quantizing the weights in the networks, such as (Han et al. 2015; Lin et al. 2017; Dong, Chen, and Pan 2017; Hubara et al. 2016; Lin, Zhao, and Pan 2017). However, these methods are not flexible during training and they do not perform well in terms of accuracy. The second category of methods are proposing more efficient structures to remove the redundancy of the parameters. One of the most popular structures is the branchy network, such as the latest MobileNet family (Howard et al. 2017; Sandler et al. 2018) and IGCV family (Xie et al. 2018; Sun et al. 2018) networks. These branchy architectures are implemented by "group convolutions" and also called group-conv mobile CNNs.



### Group convolutions

It is generally known that a regular convolutional kernel has spatial extent and channel extent, where the former value is always much smaller than the latter, for instance, a kernel with a size of "$3 \times 3 \times 32 \times 32$". Therefore, "group convolutions" is proposed to remove the redundancy from the channel extent. It equally divides the input channels and the convolutional filters into several groups, and in every group , it performs the corresponding filters. "Group convolutions" can effectively reduce parameters and computations. However, there are no interactions among the groups. This leads to the less accurate approximations of the regular kernel. To resolve this issue, the "shuffle channel" was proposed in ShuffleNet (Zhang et al. 2017).

### Shuffle channel operation

Shuffle channel operation was used between two "group convolutions" to shuffle the out feature maps from all the groups along the channel extent. The general process is shown in Figure 1. Suppose it first splits the input channels to $G$ parts and every part has $C$ channels. Then, every part is fed to the first corresponding group convolutions ($G$ groups). Next, all the output feature maps are concatenated together and permuted to be divided into $C$ partitions. Finally, another "group convolutions" with $C$ groups are performed separately on $C$ partitions and concatenate all the output feature maps. Although this operation can remedy the no connections for different groups along the channel extent, it still needs the convolutional kernels with a spatial size of "$3 \times 3$", which is redundant in the spatial dimensions .

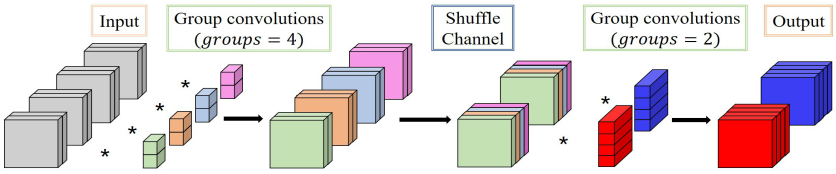



Figure 1: General combinational structure of "group convolutions" and "shuffle channel" operations.

### Motivation of super sparse convolutional kernel

Considering these issues, a new viewpoint is put forward in this paper for the first time that the feature maps' general geometric characteristics and the information's differentiation from different channels can be preserved through the selection of one pixel from every channel at different spatial locations. Accordingly, the introduced super sparse convolutional (SSC) kernel in this paper should select the pixels from the feature maps and perform operations on them by satisfying the above requirements. Consequently, a 2 dimensional spatial convolutional kernel is used to dilute and place its parameters into 3 dimensions at the same spatial positions, which is called super sparse convolutional kernel. Figure 2 illustrates the comparisons of the regular convolutional kernel and the SSC kernel. It is notable that in this 3 dimensional kernel, every spatial kernel just has one parameter. The SSC operations can be easily and more efficiently implemented by "shift"(proposed in Shift-Nets (Wu et al. 2017)) and "grouped point-wise" convolutional operations without any spatial kernels (*e.g.*, "$3 \times 3$"). Shift-Net only removes the spatial's redundancy, and group-conv mobile CNNs only removes the channel's redundancy. However, our SSC is the first method to remove the redundancy from both the spatial and channel extents at the same time. Furthermore, the SSC kernels can keep the spatial geometric characteristic and maintain the channels' differentiation based on its none-zero values' locations. All of these factors lead to largely decreasing the model's size than the other popular CNNs without performance loss and avoiding the over-fitting more effectively than the other state-of-the-art mobile CNNs.

## Super sparse convolutional neural networks

### Super sparse convolutional kernel

In Figure 2b, a SSC kernel can be seen as diluting a two dimensional spatial kernel (called the *basic kernel*) into a three dimensional kernel. In this way, the spatial non-zero locations are kept the same with the *basic kernel*, which can preserve the general geometric characteristics. And the diluting process can maintain the features' differences along the channel extent. Suppose the 4 dimensional regular convolutional kernel is $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{k\times k\times C\times D}$, where $k \times k$ indicates the spatial kernel size. $C$ and $D$ refer to the number of input and output channels respectively. The SSC kernel is denoted by $\boldsymbol{\mathcal{S}} \in \mathbb{R}^{k\times k\times C\times D}$. Then, $\boldsymbol{\mathcal{S}}$ can be extended from the *basic kernel* $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{k\times k\times D}$. It is noteworthy that in a SSC kernel, $C = k \times k$. Therefore, the definition of SSC can be formulated as below:

$$\boldsymbol{\mathcal{S}}^{\boldsymbol{x,y}}_{\boldsymbol{i,j}} = \begin{cases} \boldsymbol{\mathcal{W}}^{x,y}_{j}, & i = x \times k + y \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Where $x$, $y$ indicate the spatial location and $x$, $y \in \{0, 1, 2, ..., k-1\}$. Besides this, $i \in \{0, 1, 2, ..., C-1\}$ and $j \in \{0, 1, 2, ..., D-1\}$.

### Implementation of the SSC operations

Given an input tensor $\boldsymbol{\mathcal{I}} \in \mathbb{R}^{w\times h\times C}$, which is conducted by the novel SSC kernel $\boldsymbol{\mathcal{T}}$, and the relevant output tensor $\boldsymbol{\mathcal{O}} \in \mathbb{R}^{w\times h\times D}$ can be obtained as following equation:

$$\begin{aligned} \boldsymbol{\mathcal{O}}(x, y, q) = &\sum_{p=0}^{C-1}\sum_{i=0}^{k-1}\sum_{j=0}^{k-1} \boldsymbol{\mathcal{T}}(i, j, p, q) \\ &\boldsymbol{\mathcal{I}}(x + i - \delta_1, y + j - \delta_2, p). \end{aligned} \tag{2}$$

Where $\delta_1 = \delta_2 = \lfloor k/2 \rfloor$ and $q \in \{0, 1, 2, ..., D-1\}$. According to the Equation 1, the computational process of the SSC operations can be simplified by the equation below:

$$\begin{aligned} \boldsymbol{\mathcal{O}}(x, y, q) = &\sum_{p=0}^{C-1} \boldsymbol{\mathcal{T}}(i_p, j_p, p, q) \\ &\boldsymbol{\mathcal{I}}(x + i_p - \delta_1, y + j_p - \delta_2, p). \end{aligned} \tag{3}$$

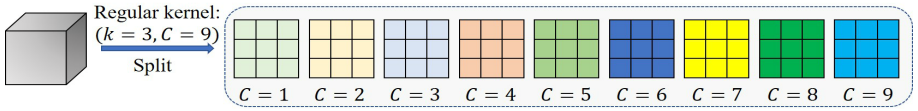
Where the $(i_p, j_p)$ is the only one nonzero parameter's spatial coordinate at the $p^{th}$ channel and $i_p \times k + j_p = p$.

From Equation 3, it is clear that an output value at the specific spatial location can be computed through the summation from $C$ times multiplication operations. Moreover, since every plane in a SSC kernel has one parameter, it can be transformed to a "point-wise" kernel. Furthermore, in order to preserve the spatial location relations in the computing process, we use a shift kernel $\boldsymbol{\mathcal{P}} \in \mathbb{R}^{k\times k\times C\times D}$ (introduced in (Wu et al. 2017)), which performs on feature maps to catch the features according to the SSC kernels' non-zero spatial locations. The shift process operation is demonstrated in Figure 3. The definition of this shift kernel is shown as below:

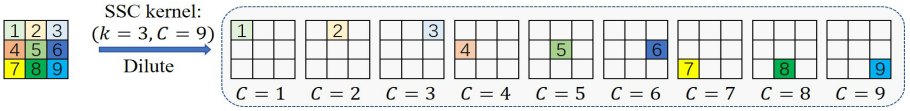
$$\boldsymbol{\mathcal{P}}^{\boldsymbol{x,y}}_{\boldsymbol{i,j}} = \begin{cases} 1, & \boldsymbol{\mathcal{S}}^{\boldsymbol{x,y}}_{\boldsymbol{i,j}} \neq 0 \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Form Equation 3 and 4, the SSC operations can be further formulated as the following equation:

$$\begin{aligned} \boldsymbol{\mathcal{O}}(x, y, q) = &\sum_{p=0}^{C-1} \boldsymbol{\mathcal{T}}(p, q)\boldsymbol{\mathcal{P}}(i_p, j_p, p, q) \\ &\boldsymbol{\mathcal{I}}(x + i_p - \delta_1, y + j_p - \delta_2, p). \end{aligned} \tag{5}$$




(a) Regular convolutional kernel: the kernel has $k \times k \times C$ parameters ($e.g.,\ k = 3,\ C = 9$), and it can be split to $C$ spatial kernels along the channel extent.




(b) SSC kernel: the kernel has $k \times k$ parameters ($e.g.,\ k = 3$), it consists of $C$ ($C = k \times k$) spatial kernels, and every spatial kernel has only one non-zero parameter.

Figure 2: Comparisons of regular convolutional and SSC kernel.

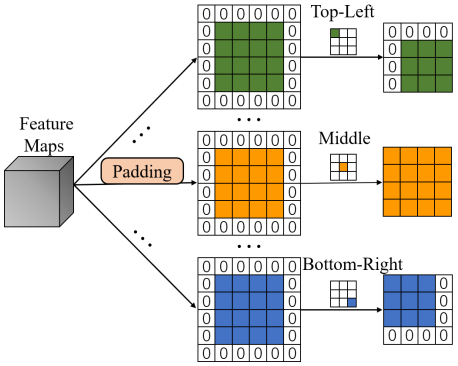



Figure 3: Shift operational process: the feature maps are first padded with zero along each side, then each padded feature map is cropped based on the relevant shift kernel's non-zero direction (the colored location in the shift kernel). In this figure, the shift kernel's spatial size is "$3 \times 3$" and the feature maps' size is "$4 \times 4$".

From Equation 5, the SSC operation can be implemented by the "shift" and "point-wise" convolutional operations, which can save much more parameters and computational Flops.

## The basic SSC module and the SSC-Nets

A " point-wise" convolutional layer is first used to project the input feature maps into a required dimensional space, which can also fuse the previous module's output features from different groups. The obtained projected feature map channels are $M$. Then, two SSC layers are equipped together with a "shuffle-channel" operation between them. Before each SSC operation, another "group point-wise" convolutional layer is utilized to force every SSC's input to have similar geometric characteristics as much as possible. Finally, similar to ResNets (Huang et al. 2016), identity mapping is also adopted in the SSC module. The detailed structure of a SSC-Net's basic module is shown in Table 1.

It is important that in the second SSC layer, we still shift the feature maps and employ "point-wise" convolutional operations on every $C$ channels. Then, the number of groups becomes $G$ ($G = d \times C$, $d$ is a positive integer). Finally, another "group point-wise" convolution with $C$ groups is utilized to merge the $G$ groups to $C$ groups. Hence, the SSC module entirely employs "point-wise" and "group point-wise" convolutions, which can largely decrease the parameters and computational Flops. The SSC-Net uses the modularized design method based on ResNet56 (Huang et al. 2016), SSC-Net also has three big blocks with different output spatial sizes (*e.g.*, $32 \times 32$, $16 \times 16$ and $8 \times 8$) and every block has $B$ basic modules. The output channels of every stage will be doubled when moving to the next stage. The other details remain the same with ResNet56 (Huang et al. 2016).

Table 1: SSC basic module. All the operations' output channels are $M$. $C = k^2$ and $G = d \times C$. $k^2$ is the SSC kernel's spatial size.

| Stage | Operation | Groups | Channels/Group |
|---|---|---|---|
| Conv1 | $1 \times 1$ | 1 | $M$ |
| Conv2 | $1 \times 1$ | $G$ | $C$ |
| $1^{st}$ SSC | *shift*<br>$1 \times 1$ | $G$<br>$G$ | $C$<br>$C$ |
| Shuffle | *shuffle* | $C$ | $G$ |
| Conv3 | $1 \times 1$ | $C$ | $G$ |
| $2^{nd}$ SSC | *shift*<br>$1 \times 1$<br>$1 \times 1$ | $G$<br>$G$<br>$C$ | $C$<br>$C$<br>$G$ |

## Analysis of SSC kernels' parameters

For a ResNet's basic module in ResNet56 (Huang et al. 2016), suppose the kernel's spatial size is "$k \times k$" and the input and output channels are $K$. Therefore, a ResNet's basic module's parameters $P_R$ are:

$$P_R = 2 \times (k \times k \times K \times K) = 2k^2K^2 \qquad (6)$$

For a SSC-Net's basic module (Table 1), suppose the input and output channels are $M$, and the groups of the first

SSC operation are $G$. Each group has $C$ channels. Thus, the number of parameters $P_S$ of a SSC-Net's basic module is:

$$\begin{aligned} P_S =& (1 \times 1 \times M \times M) + 2 \times (1 \times 1 \times M \times M/G) \\ &+ 2 \times (1 \times 1 \times M \times M/C) + (1 \times 1 \times M \times M/G) \\ =& M^2 + 3MC + 2M^2/C = (1 + 2/C)\, M^2 + 3MC \end{aligned} \tag{7}$$

From Equation 6 and 7, when $M = K$ and $C = k \times k$,

$$P_S = \frac{1}{2}\left(\frac{C+2}{C^2} + \frac{3}{M}\right) P_R \approx \frac{1}{2C} P_R. \tag{8}$$

Therefore, when $k$ is 3, $C = k \times k = 9$ and $P_S \approx \frac{1}{18} P_R$, which implies the SSC-Nets have much fewer parameters than the regular networks of the same width.

From another aspect, when the two modules have the same number of parameters ($P_S = P_R$),

$$\begin{aligned} K &= \sqrt{\frac{M}{2}\left(\frac{C+2}{C^2}M + 3\right)} \leq \frac{1}{2}\left(\frac{M}{2} + \frac{C+2}{C^2}M + 3\right) \\ &\approx \left(\frac{1}{4} + \frac{1}{2C}\right) M \approx \frac{1}{4} M, \end{aligned} \tag{9}$$

this indicates SSC-Net is at least **4X** wider than the regular model with the same parameters. Hence, SSC-Net can process and produce much more features. In addition, since the number of Flops is "$w \times h$" (indicating the width and height of the feature map) times of the parameters, the SSC-Nets also have notable advantages when comparing the Flops.

## Experimental result and analysis

### Datasets

This paper's goal is to design mobile models to reduce the number of parameters and computational complexity without any loss in performance and overcome the over-fitting problem on the less abundant datasets. As illustrated in (Chrabaszcz, Loshchilov, and Hutter 2017), since the low resolution ImageNet datasets' spatial informations are much less abundant than the original ImageNet datasets, the low resolution ImageNet can make the model easily over-fitting and become more challenging than the original ImageNet. Therefore, we select the benchmark low resolution ImageNet and CIFAR (Krizhevsky and Hinton 2009) datasets.

**CIFAR datasets** (Krizhevsky and Hinton 2009) have a small number of images including CIFAR-10 and CIFAR-100. They both have $60,000$ colored nature scene images in total and the images' size is $32 \times 32$. There are $50,000$ images for training and $10,000$ images for testing in 10 and 100 classes. Data augmentation is the same with the common practice in (He et al. 2016a; Huang et al. 2016; Larsson, Maire, and Shakhnarovich 2016).

**Low resolution ImageNet datasets** (Chrabaszcz, Loshchilov, and Hutter 2017) are the down-sampled variants of ImageNet (Deng et al. 2009) and contain the same number of classes and images of ImageNet. There are three versions in total: ImageNet-$64 \times 64$, ImageNet-$32 \times 32$ and ImageNet-$16 \times 16$, which indicates the down-sampled images' sizes are respectively $64 \times 64$, $32 \times 32$ and $16 \times 16$. In order to keep the same spatial size with CIFAR datasets, we perform SSC-Nets on ImageNet-$32 \times 32$. The augmentation of this dataset is the same with (Chrabaszcz, Loshchilov, and Hutter 2017).

### Initialization and hyper-parameters

The different versions of SSC-Nets, ResNet and Shift-Nets are respectively indicated by SSC-Nets-$B$-$G$, ResNets-$B$-$\varepsilon$ and Shift-Nets-$B$-$\varepsilon$, where $B$ represents the number of basic modules in a big block. $G$ denotes the number of groups in the first block. $\varepsilon$ is the expansion parameter (Wu et al. 2017). ResNets and Shit-Nets will have different model sizes by toggling $\varepsilon$.

The almost identical weight initialization and optimization configuration introduced in ResNet are adopted in SSC-Nets. The mini-batch size is set to 128. The SGD method and Nesterov momentum (Sutskever et al. 2013) are utilized in the optimization. Where the momentum is $0.9$. On CIFAR, the training epoch is set to 160, and the optimization starts from the initial learning rate with $0.1$, which is divided by 10 at the $80^{th}$ and $120^{th}$ epoch. For ImageNet, the training epoch is $40$. The learning rate starts from $0.01$ and is divided by 10 every 10 epoches. Finally, the weight decay is set to $0.0002$ and $0.0001$ on CIFAR and ImageNet, respectively.

### Parameter comparisons with approximately the same accuracy

SSC-Nets and Shift-Nets both employ "point-wise" convolutions entirely, where SSC-Nets are implemented based on ResNet. Consequently, SSC-Nets are compared with Shift-Nets and ResNets. We first compare the models' sizes under the approximately the same accuracy. The final number of parameters is shown in Table 2. The results of ResNets and Shift-Nets are reported in (Wu et al. 2017), and the value in the "params" column indicates a rough level of these two models' parameters, since some models' parameters and detailed structures are not illustrated in (Wu et al. 2017). It is apparent that SSC-Nets have much fewer parameters than the ResNets and Shift-Nets. SSC-Nets-2-9 reduces about **3.5X** parameters compared with ResNet-18-6 and Shift-Net-18-6. In addition, SSC-Net-3-9 also reduces about **3.1X** parameters compared with ResNet-18-9 and Shift-Net-18-9. What is more, the results show that SSC-Nets can even achieve better performances than the compared models. For instance, SSC-Net-3-9 with $0.56M$ parameters improves the CIFAR10 accuracy by $1.14\%$ than Shift-Net-18-9. Additionally, compared with the best CIFAR10 accuracy $93.17\%$ achieved by the Shift-Net-18-6 with $1.18M$ parameters, Shift-Net-18-9 with $1.76M$ parameters decreases the accuracy, which is caused by the over-fitting problem. However, the larger model sized SSC-Nets-3-9 still outperforms SSC-Nets-2-9 through increasing the CIFAR10 accuracy by $0.69\%$. The experimental results show that the SSC-Nets can avoid this problem effectively.

We also make some accuracy comparisons with SSC-Nets possessing **1.5X** fewer parameters than ResNets and Shift-Nets (shown in Table 3). The results show that SSC-Nets can

Table 2: Comparisons of parameters with approximately the same accuracy (%) on CIFAR.

| Method | Params ($M$) | CIFAR10 | CIFAR100 |
|---|---|---|---|
| ResNet-18-6 | 1.18 | 79.02 | 68.87 |
| Shift-Net-18-6 | 1.18 | 93.17 | 72.56 |
| SSC-Net-2-9 | **0.34** | **93.24** | **72.62** |
| ResNet-18-9 | 1.72 | 92.46 | 72.11 |
| Shift-Net-18-9 | 1.76 | 92.79 | 74.10 |
| SSC-Net-3-9 | **0.56** | **93.93** | **74.32** |

Table 3: Accuracy (%) of SSC-Nets with **1.5X** fewer parameters compared with ResNets and Shit-Nets on CIFAR.

| Method | Params ($M$) | CIFAR10 | CIFAR100 |
|---|---|---|---|
| ResNet-9-6 | 0.58 | 89.89 | 67.45 |
| Shift-Net-9-6 | 0.58 | 92.69 | 72.13 |
| SSC-Net-2-9 | **0.34** | **93.24** | **72.62** |
| ResNet-9-9 | 0.85 | 92.01 | 69.27 |
| Shift-Net-9-9 | 0.87 | 92.74 | 73.64 |
| SSC-Net-3-9 | **0.56** | **93.93** | **74.32** |
| ResNet-18-9 | 1.72 | 92.46 | 72.11 |
| Shift-Net-18-9 | 1.76 | 92.79 | 74.10 |
| SSC-Net-6-9 | **1.15** | **95.14** | **75.99** |

consistently outperform ResNets and Shift-Nets. Compared with the largest ResNet and Shift-Net models, SSC-Net-6-9 achieves much better accuracies of 95.14% and 75.99% on CIFAR10 and CIFAR100 with only $1.15M$ parameters.

Finally, the comparisons of parameters using Top-1 accuracy are performed on ImageNet-32 $\times$ 32 listed in Table 4. The experimental results show that SSC-Nets achieve a slightly better performance by utilizing **2X** or **2.5X** fewer parameters than ResNets. It proves that SSC-Nets can also perform effectively when facing more challenging datasets.

## Accuracy comparisons with approximately the same number of parameters

In order to test the performance when the model size of SSC-Nets increases, the accuracy comparisons are made by designing different SSC-Nets, whose parameters are approximately the same with the relevant ResNet and Shift-Net. The final experimental results are shown in Table 5. It is notable that all SSC-Nets outperform the compared networks at their corresponding sized level. Especially, the the accuracies obtained by SSC-Net-3-9, where it increases by 1.24% and 2.19% on CIFAR10 and CIFAR100 compared with Shift-Net-9-6. The accuracies obtained by SSC-Net-6-9 increase by 1.97% and 3.43% on CIFAR10 and CIFAR100 compared with Shift-Net-18-6. Finally, compared with Shift-Net-18-9, the largest model SSC-Net-9-9 improves the CIFAR10 and CIFAR100 accuracies by 1.95% and 2.50%, respectively.

Additionally, on ImageNet-32 $\times$ 32 datasets (see Table 6), SSC-Nets can also significantly improve the performances.

In order to intuitively express the superiorities of SSC-Nets, Figure 4 and Figure 5 show the performances achieved by different models with various parameters and computational complexities, respectively. It is clear that SSC-Net is more efficient than the other models. It can also achieve much better results with much fewer parameters and Flops.

Table 4: Parameter comparison with approximately the same Top-1 accuracy (%) on Imagenet-32 $\times$ 32.

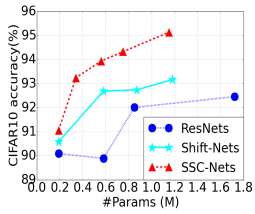
| Method | Params ($M$) | Top-1 | Reduction |
|---|---|---|---|
| ResNet-4-4 | 1.6 | 43.08 | |
| SSC-Net-3-9 | **0.8** | **43.88** | **2X** |
| ResNet-4-8 | 3.5 | 48.94 | |
| SSC-Net-6-9 | **1.4** | **49.23** | **2.5X** |

Table 5: Accuracy (%) on CIFAR datasets with approximately the same number of parameters.

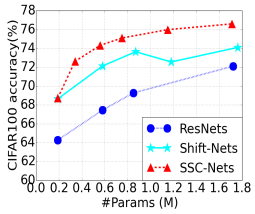
| Method | Params ($M$) | CIFAR10 | CIFAR100 |
|---|---|---|---|
| ResNet-3-6 | 0.19 | 90.09 | 64.27 |
| Shift-Net-3-6 | 0.19 | 90.59 | 68.64 |
| ResNet-18-1 | 0.21 | 76.82 | 60.44 |
| Shift-Net-18-1 | 0.20 | 90.34 | 67.84 |
| SSC-Net-1-9 | 0.19 | **91.06** | **68.73** |
| ResNet-9-6 | 0.58 | 89.89 | 67.45 |
| Shift-Net-9-6 | 0.58 | 92.69 | 72.13 |
| ResNet-18-3 | 0.59 | 74.30 | 66.61 |
| Shift-Net-18-3 | 0.59 | 91.98 | 71.83 |
| SSC-Net-3-9 | 0.56 | **93.93** | **74.32** |
| ResNet-18-6 | 1.18 | 79.02 | 68.87 |
| Shift-Net-18-6 | 1.18 | 93.17 | 72.56 |
| SSC-Net-6-9 | 1.15 | **95.14** | **75.99** |
| ResNet-18-9 | 1.72 | 92.46 | 72.11 |
| Shift-Net-18-9 | 1.76 | 92.79 | 74.10 |
| SSC-Net-9-9 | 1.71 | **94.74** | **76.60** |

Table 6: Accuracy (%) on ImageNet-32 $\times$ 32 with approximately the same number of parameters.

| Method | Params ($M$) | Top-1 | Top-5 |
|---|---|---|---|
| ResNet-4-2 | 1.0 | 39.55 | 65.16 |
| Shift-Net-4-2 | 1.0 | 41.47 | 67.44 |
| SSC-Net-4-9 | 1.0 | **45.91** | **70.93** |
| ResNet-4-4 | 1.6 | 43.08 | 69.08 |
| Shift-Net-4-4 | 1.5 | 44.43 | 70.03 |
| SSC-Net-6-9 | 1.4 | **49.23** | **73.81** |
| ResNet-4-8 | 3.5 | 48.94 | 73.92 |
| Shift-Net-4-8 | 3.5 | 51.85 | 75.88 |
| SSC-Net-4-18 | 3.5 | **53.46** | **77.25** |

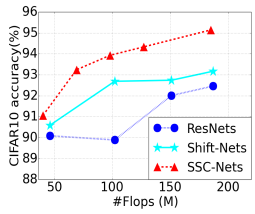



(a) CIFAR10 acc

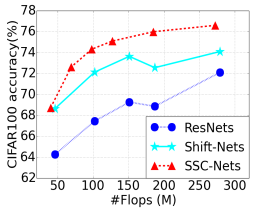



(b) CIFAR100 acc

Figure 4: Accuracy vs. parameters tradeoff.




(a) CIFAR10 acc




(b) CIFAR100 acc

Figure 5: Accuracy vs. Flops tradeoff.

## Comparisons with state-of-the-art mobile CNNs

Finally, in order to sufficiently explore the SSC-Nets' performances, various SSC-Nets are compared with the other state-of-the-art mobile models. On the ImgeNet-32 $\times$ 32, Table.7 shows that SSC-Net achieves the best Top-1 and Top-5 accuracy than the other mobile models. It also proves that reducing spatial redundancy (SSC-Net and Shift-Net) can avoid over-fitting better than reducing channel redundancy (IGCV3-D 1.0), and SSC performs best because it reducing both the spatial and channel redundancy. Hence, the SSC-Net can remove the redundancy more thoroughly than the other mobile CNNs. On the CIFAR, from the Table. 8, it shows that the DenseNet is a optimal structure, because it uses the dense connections and bottle-neck to decrease the number of parameters and improve the performance. But, DenseNets are not very practical to be deployed on the mobile devices compared with the other group-conv mobile models, since the dense connections bring large memory storage in practice. For our SSC-Nets, it is clear that they can produce much better performances than the other group-conv mobile networks on CIFAR-10. As for the slightly difficult CIFAR 100, since SSC is entirely implemented by the point-wise convolutions in practice, SSC-Net doesn't achieve the best accuracy, but, it can also meet the general accuracy requirement.

## Further investigation of the SSC kernel

**The sparse property of the SSC Kernel** In order to intuitively observe the sparsity of the SSC kernel. According to (Ioannou et al. 2017), the filters' relationships from two layers can be obtained through calculating the inter-covariances of the two successive layers' response output channels. The corresponding inter-covariances from different stages of the Shift-Net and SSC filters are shown in Figure 6. The smaller the covariance, the sparser the filters are. From the Figure 6d, 6e and 6f, the SSC filters obtain an evident block diagonal sparsity, which implies the 4 dimensional SSC kernel is very sparse. Furthermore, in every group, all the 3 dimensional SSC filters from the two successive layers are clearly clustered in a block with strong relationships, which effectively demonstrates that each 3 dimensional SSC filter can retrieve the specific geometric characteristics in its corresponding group. Consequently, the SSC kernel is more targeted and can process more information efficiently. This contributes to obtaining better features through its super sparse structures and effectively preventing any over-fitting.

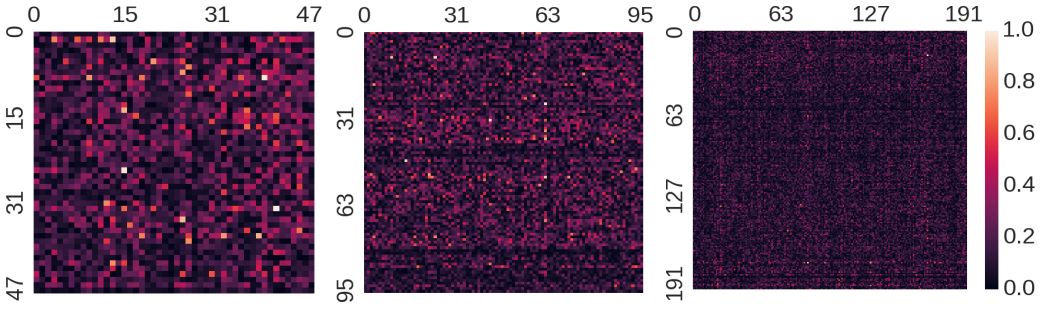
Table 7: Comparisons of accuracy (%) with the latest state-of-the-art mobile models on Imagenet-32 $\times$ 32.

| Method | Params ($M$) | Top-1 | Top-5 |
|---|---|---|---|
| ResNet-4-8 | 3.5 | 48.94 | 73.92 |
| MobileNetV2 | 3.5 | 48.98 | 73.82 |
| IGCV-D 1.0$\times$ | 3.5 | 49.40 | 74.04 |
| Shift-Net-4-8 | 3.5 | 51.85 | 75.88 |
| SSC-Net-4-18 | 3.5 | **53.46** | **77.25** |

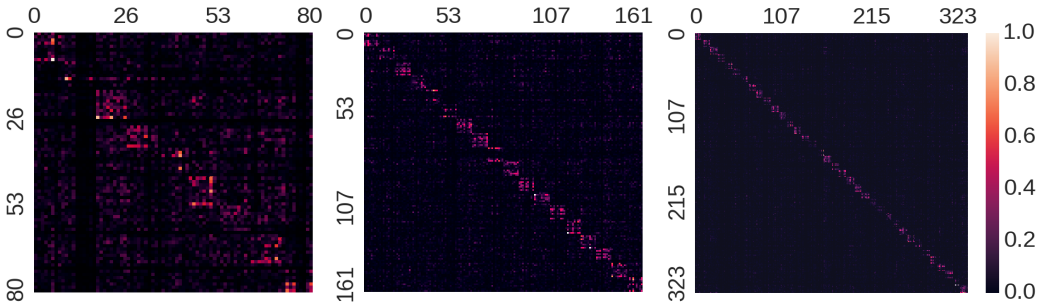
Table 8: Accuracy (%) comparisons to other state-of-the-art small and medium sized architectures on CIFAR.

| Method | Params ($M$) | CIFAR10 | CIFAR100 |
|---|---|---|---|
| Swapout (Singh, Hoiem, and Forsyth 2016) | 1.1 | 93.42 | 74.14 |
| DenseNet (Huang et al. 2017b) | 1.0 | 94.76 | 75.58 |
| DenseNet-BC ($k = 12$) (Huang et al. 2017b) | 0.8 | 95.49 | 77.73 |
| ResNet (Huang et al. 2017a) | 1.7 | 94.48 | 71.98 |
| ResNet(pre-act) (He et al. 2016b) | 1.7 | 94.54 | 75.67 |
| DFM-MP1 (Zhao et al. 2016) | 1.7 | 95.06 | 75.54 |
| MobileNetV2 (Sun et al. 2018) | 2.3 | 94.56 | 77.09 |
| IGCV2*-$C416$ (Xie et al. 2018) | 0.7 | 94.51 | 77.05 |
| IGCV2 (Sun et al. 2018) | 2.3 | 94.76 | 77.45 |
| IGCV3-D 0.7$\times$ (Sun et al. 2018) | 1.2 | 94.92 | 77.83 |
| IGCV3-D 1.0$\times$ (Sun et al. 2018) | 2.4 | 94.96 | 77.95 |
| Shift-Net-18-6 | 1.2 | 93.17 | 72.56 |
| SSC-Net-6-9 | 1.2 | **95.14** | 75.99 |
| SSC-Net-3-18 | 2.1 | 94.55 | 77.13 |
| SSC-Net-4-18 | 2.8 | 94.95 | 77.67 |

**Comparisons of sparsity ability** Based on the concept of SSC, SSC-Net is more sparser in the form because of the virtual zero parameters existing in the SSC filter's structure. However, in order to further explores this sparse structure's effect on the SSC's real parameters, Figure 7 shows the cumulative distribution probabilities of the convolutional weights from ResNets, ShiftNets and SSC-Nets with almost the same model size level on CIFAR 10. Figure 7a, 7b and 7c showed all the SSC-Nets' weights are closer to zero than other models with the same number of parameters. This re-

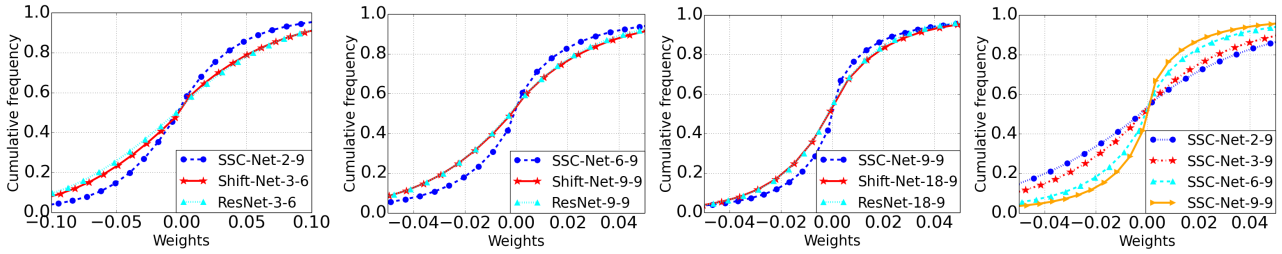



(a) Stage 1 of Shift-Nets (b) Stage 2 of Shift-Nets (c) Stage 3 of Shift-Nets




(d) Stage 1 of SSC-Nets (e) Stage 2 of SSC-Nets (f) Stage 3 of SSC-Nets

Figure 6: Inter covariances of SSC-Nets and Shift-Nets.




(a) On $0.19M$ params (b) On $0.87M$ params (c) On $1.76M$ params (d) Various SSC-Nets

Figure 7: Comparisons of sparsity ability on CIFAR10.

veals the real nature of SSC's sparsity ability and the reason why sparse models can avoid over-fitting better on the less abundant dataset. Additionally, when the model size becomes larger, Shift-Net's sparsity ability decreases gradually until almost to same extent with ResNet. However, from Figure 7d, when the model size increases, SSC-Net's sparsity ability is also improved. This proves that SSC can combat over-fitting more effectively on the less abundant datasets.

## Conclusion

A novel super sparse convolutional kernel (SSC) is proposed in this paper. The SSC kernel is much sparser than the traditional convolutional kernel, since it is the first method to reduce the parameters' redundancy from both the spatial and the channel extents at the same time. Additionally, the retrieved features by the SSC kerenl can preserve the general characteristics of the geometric and the channels' differences and the computational Flops can be also largely decreased. Finally, it is effectively implemented by "shift" and "group point-wise" convolutional operations. Experimental results show that SSC-Nets can effectively decrease the model's size without any performance losses and address the over-fitting better than the other state-of-the-art mobile networks on the more challenging less abundant databases.

## Acknowledgments

This work is supported by NSFC fund (61332011), Shenzhen Fundamental Research fund (JCYJ20170811155442454), and Medical Biometrics Perception and Analysis Engineering Laboratory, Shenzhen, China.

## References


Chrabaszcz, P.; Loshchilov, I.; and Hutter, F. 2017. A downsampled variant of imagenet as an alternative to the cifar datasets. *arXiv preprint arXiv:1707.08819*.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 248–255. IEEE.

Dong, X.; Chen, S.; and Pan, S. 2017. Learning to prune deep neural networks via layer-wise optimal brain surgeon. In *Advances in Neural Information Processing Systems*, 4860–4874.

Han, S.; Pool, J.; Tran, J.; and Dally, W. 2015. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems*, 1135–1143.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016a. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016b. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, 630–645. Springer International Publishing.

Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; and Adam, H. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR* abs/1704.04861.

Huang, G.; Sun, Y.; Liu, Z.; Sedra, D.; and Weinberger, K. 2016. Deep networks with stochastic depth. In *European Conference on Computer Vision*, 646–661. Springer International Publishing.

Huang, G.; Li, Y.; Pleiss, G.; Liu, Z.; Hopcroft, J. E.; and Weinberger, K. Q. 2017a. Snapshot ensembles: Train 1, get m for free. *arXiv preprint arXiv:1704.00109*.

Huang, G.; Liu, Z.; Weinberger, K. Q.; and van der Maaten, L. 2017b. Densely connected convolutional networks. *arXiv preprint arXiv:1608.06993v4*.

Hubara, I.; Courbariaux, M.; Soudry, D.; El-Yaniv, R.; and Bengio, Y. 2016. Binarized neural networks. In *Advances in neural information processing systems*, 4107–4115.

Ioannou, Y.; Robertson, D.; Cipolla, R.; Criminisi, A.; et al. 2017. Deep roots: Improving cnn efficiency with hierarchical filter groups.

Krizhevsky, A., and Hinton, G. 2009. Learning multiple layers of features from tiny images.

Larsson, G.; Maire, M.; and Shakhnarovich, G. 2016. Fractalnet: Ultra-deep neural networks without residuals. *arXiv preprint arXiv:1605.07648*.

Lin, J.; Rao, Y.; Lu, J.; and Zhou, J. 2017. Runtime neural pruning. In *Advances in Neural Information Processing Systems*, 2178–2188.

Lin, X.; Zhao, C.; and Pan, W. 2017. Towards accurate binary convolutional neural network. In *Advances in Neural Information Processing Systems*, 344–352.

Sandler, M.; Howard, A. G.; Zhu, M.; Zhmoginov, A.; and Chen, L. 2018. Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. *CoRR* abs/1801.04381.

Singh, S.; Hoiem, D.; and Forsyth, D. 2016. Swapout: Learning an ensemble of deep architectures. In *Advances in neural information processing systems*, 28–36.

Sun, K.; Li, M.; Liu, D.; and Wang, J. 2018. Igcv3: Interleaved low-rank group convolutions for efficient deep neural networks. *CoRR* abs/1806.00178.

Sutskever, I.; Martens, J.; Dahl, G.; and Hinton, G. 2013. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, 1139–1147.

Wu, B.; Wan, A.; Yue, X.; Jin, P.; Zhao, S.; Golmant, N.; Gholaminejad, A.; Gonzalez, J.; and Keutzer, K. 2017. Shift: A zero flop, zero parameter alternative to spatial convolutions. *arXiv preprint arXiv:1711.08141*.

Wu, J.; Li, D.; Yang, Y.; Bajaj, C.; and Ji, X. 2018. Dynamic sampling convolutional neural networks. *arXiv preprint arXiv:1803.07624*.

Xie, G.; Wang, J.; Zhang, T.; Lai, J.; Hong, R.; and Qi, G. 2018. IGCV2: interleaved structured sparse convolutional neural networks. *CoRR* abs/1804.06202.

Zhang, X.; Zhou, X.; Lin, M.; and Sun, J. 2017. Shufflenet: An extremely efficient convolutional neural network for mobile devices. *arXiv preprint arXiv:1707.01083*.

Zhao, L.; Wang, J.; Li, X.; Tu, Z.; and Zeng, W. 2016. Deep convolutional neural networks with merge-and-run mappings. *arXiv preprint arXiv:1611.07718*.