INTEGRATING LARGE CIRCULAR KERNELS INTO CNNS THROUGH NEURAL ARCHITECTURE SEARCH

Kun He[*], **Chao Li**[†], **Yixiao Yang**
School of Computer Science and Technology,
Huazhong University of Science and Technology
Wuhan, 430074, China
{brooklet60, d201880880, m201973180}@hust.edu.cn

Gao Huang
Department of Automation,
Tsinghua University
Beijing, 100084, China
gaohuang@tsinghua.edu.cn

John E. Hopcroft
Computer Science Department,
Cornell University
Ithaca, USA
jeh@cs.cornell.edu

ABSTRACT

The square kernel is a standard unit for contemporary CNNs, as it fits well on the tensor computation for convolution operation. However, the retinal ganglion cells in the biological visual system have approximately concentric receptive fields. Motivated by this observation, we propose to use circular kernel with a concentric and isotropic receptive field as an option for the convolution operation. We first propose a simple yet efficient implementation of the convolution using circular kernels, and empirically show the significant advantages of large circular kernels over the counterpart square kernels. We then expand the operation space of several typical Neural Architecture Search (NAS) methods with the convolutions of large circular kernels. The searched new neural architectures do contain large circular kernels and outperform the original searched models considerably. Our additional analysis also reveals that large circular kernels could help the model to be more robust to the rotated or sheared images due to their better rotation invariance. Our work shows the potential of designing new convolutional kernels for CNNs, bringing up the prospect of expanding the search space of NAS with new variants of convolutions.

1 INTRODUCTION

The square convolution kernel has been regarded as the standard and core unit of Convolutional Neural Networks (CNNs) since the first recognized CNN of *LeNet* proposed in 1989 (LeCun et al., 1998), and especially after *AlexNet* (Krizhevsky et al., 2012) won the ILSVRC (ImageNet Large Scale Visual Recognition Competition) in 2012. Since then, various variants of convolution kernels have been proposed, including separable convolution (Chollet, 2017), dilated convolution (Yu & Koltun, 2016), deformable convolution (Jeon & Kim, 2017; Dai et al., 2017; Zhu et al., 2019; Gao et al., 2020), *etc*. Inspired by the fact that the retinal ganglion cells in the biological visual system have approximately concentric receptive fields (RFs) (Hubel & Wiesel, 1962; Simoncelli & Olshausen, 2001; Mutch & Lowe, 2008), we propose the concept of circular kernels for the convolution operation[1]. As shown in Fig. 1, a $K \times K$ circular kernel is defined as a kernel that evenly samples K^2 pixels on the concentric circles to form a circular receptive field.

Besides the similarity to biological RFs, we observe that the circular kernel provides many advantages over the square kernel. First, the receptive field of a kernel is traditionally expected to be isotropic to

[*]The first three authors contribute equally.
[†]Corresponding author.
[1]Code: https://github.com/JHL-HUST/CircularKernel.

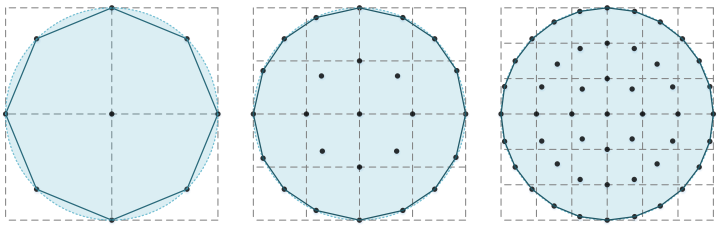


Figure 1: The receptive fields and sampling points of circular kernels in size $k \in \{3, 5, 7\}$. The intersection of dashed lines are the sampling points of square kernels in size $k \in \{3, 5, 7\}$. All circular receptive fields are concentric and approximately isotropic. A larger circular kernel has a more round receptive field

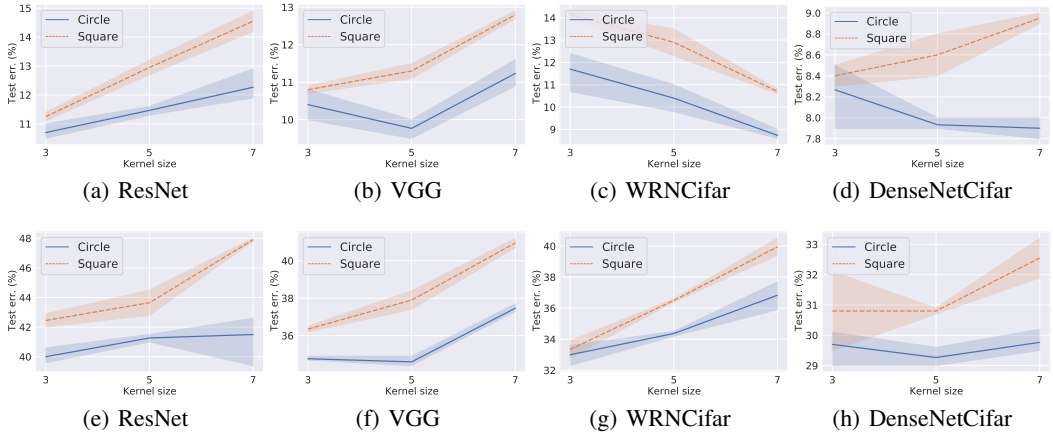
fit thousands of uncertain symmetric orientations of the input feature maps, either globally or locally. An *isotropic* kernel means the kernel samples evenly in different directions of the RFs. The circular kernel is roughly isotropic and rotation-invariant, whereas a square kernel is symmetric only in a few orientations. Second, Luo *et al*. (Luo et al., 2016) indicate that the effective RF of a square kernel has a Gaussian distribution which is in a nearly circular shape. It indicates that the meaningful weights are sparse at the four corners of large square kernels or stacked 3×3 square kernels. Compared to pruning these diluted parameters during the fine-tuning stage (Han et al., 2015), directly constructing kernels with the same shape of effective RFs is probably more effective.

One cornerstone of the rationality of employing circular kernels is the isotropic property of circles. However, a 3×3 circular kernel is not really in circular shape as it only samples nine pixels with a similar arrangement to the square kernel. If we build the circular kernels in larger kernel size, as illustrated in Fig. 1, we can see that a larger circular kernel has a more round receptive field and is more distinct from the corresponding square kernel. Our follow-up experiments also demonstrate that the circular kernels exhibit significant advantages over the square kernels on larger kernel sizes.

The 3×3 square kernels have become the mainstream of the CNN units since the work of VGG (Simonyan & Zisserman, 2015) suggests that a larger square kernel could be substituted by several 3×3 square kernels utilizing fewer parameters. In recent years, however, the functions of larger square kernels have been considered underestimated, as almost all the powerful models generated by Neural Architecture Search (NAS) (Zoph et al., 2018; Liu et al., 2018a; Xu et al., 2020; Nayman et al., 2019) contain large square kernels, and many manually designed neural architectures also contain large square kernels (He et al., 2016; Peng et al., 2017; Li et al., 2019). The recent success of ConvNeXt (Liu et al., 2022) over Swin transformer (Liu et al., 2021) also shows that increasing the kernel size can significantly improve the performance. Compared to small kernels, large kernels have received insufficient attention despite their vast range of applications. Hence, we introduce convolutions with large circular kernels as an option for the CNN units, especially for NAS.

The mainstream CNNs are manually developed and optimized on the 3×3 square kernels. So variants of kernels encounter significant resistance to outperform 3×3 square kernels on the existing popular architectures. NAS aims to design a neural architecture that performs best under limited computing resources in an automated manner (Ren et al., 2020). It creates a level playing field for different types of operations in the search space. In manually designed architectures, the network typically contains the same unit for all layers (e.g., 3×3 square kernels) because it is hard to arrange them in different layers appropriately if we have several different units. For the convolution operation, although it is hard to substitute the standard convolution with the variants in all layers of the typical manual architectures, NAS enables the variants to exist in the proper position as a part of the overall network. Then some special variants are likely to outperform the standard operations if being located in the right position. Consequently, although existing NAS methods have achieved superior performance, their search space that only contains popular operations used in manual architectures seems conservative.

In this work, we propose to use the convolution of large circular kernels with a concentric and isotropic receptive field as an option for the search space of NAS methods. As shown in Fig. 2, by simply substituting the 3×3 square kernels with 3×3 circular kernels in manual CNN architectures, the performance of the modified network after training could be on par with the original network,



(a) ResNet (b) VGG (c) WRNCifar (d) DenseNetCifar

(e) ResNet (f) VGG (g) WRNCifar (h) DenseNetCifar

Figure 2: Test error (%) of the baselines with square kernels (dashed lines) and the corresponding circular kernel (solid lines) versions in kernel size $k \in \{3, 5, 7\}$ on CIFAR-10 (*top*) and CIFAR-100 (*bottom*). For a fair comparison, we use the original data without data augmentation (best viewed in color)

even though the original one is designed and hence optimized manually on square kernels. Moreover, with the increment of kernel size on the modified models adapted on typical manual models, although the overall performance declines for both square kernels and circular kernels, the circular kernels actually exhibit significantly increasing advantages over the counterpart square kernels.

Our preliminary experiments inspire us to expand the search space of NAS methods with convolutions of large circular kernels. In this way, NAS may enable the convolutions of large circular kernels to be located in **proper position** to outperform the standard operations. Note that many works have shown that simply enlarging the operation space could be detrimental to the final results (Yu et al., 2020b; Zhang et al., 2021; Ci et al., 2020). Hence, our method of expanding the operation space with convolutions of large circular kernels is beneficial for the NAS.

Our main contributions are as follows:

- We first propose to use the circular kernel with a concentric and isotropic receptive field as an option for the convolution operation, especially for the convolution of kernels with larger kernel size.
- We propose a simple yet efficient implementation for the convolution of the circular kernel, enabling it to work together seamlessly with any CNNs with little extra time. We also show that the circular kernel has an optimization path different from that of the square kernel.
- We propose to use the convolution of large circular kernels as an option for the search space of NAS methods because they enable large circular kernels to be located in proper position, and our experiments show that the searched architectures contain large circular kernels and outperform the original ones containing only square kernels.
- Our study reveals an important phenomenon that the variants, that perform averagely in manual architecture because of inherent mode of thinking, probably have extraordinary performance in neural architecture search. We emphasize that the search space can be expanded as new designs emerge.

2 Related Works

Understanding and exploring the convolution units has always been an essential topic in the field of deep learning. In this section, we review the previous primary efforts on the design of convolution kernel and CNN architecture, then show how our work differs.

Convolution Kernel Design. The grouped convolution uses a group of convolutions (multiple kernels per layer) to allow the network to train over multi-GPUs (Krizhevsky et al., 2012). The

depthwise separable convolution decomposes a standard convolution into a depthwise convolution followed by a pointwise convolution (Chollet, 2017). The spatially separable convolution decomposes a $K \times K$ square kernel into two separate units, a $K \times 1$ kernel and a $1 \times K$ kernel (Mamalet & Garcia, 2012). The dilated convolution is a type of convolution that "inflate" the kernel by inserting holes between the kernel elements (Yu & Koltun, 2016). All the above variants consider large kernels but inherit the square kernel in general.

In contrast, the deformable convolution (Dai et al., 2017; Zhu et al., 2019) allows the shape of the receptive field to be learnable based on the input feature maps to provide flexibility, but it needs to take considerable extra parameters and computation overhead. Similarly, the deformable kernel (Gao et al., 2020) resamples the original kernel space while keeping the receptive field unchanged. There are also many interesting variants with special shapes, including quasi-hexagonal convolution (Sun et al., 2016), blind-spot convolution (Krull et al., 2019), asymmetric convolution (Ding et al., 2019), *etc*. The above variants change the kernel shape but ignore large kernels.

In the early stage of CNN design, the kernel size gradually evolves from large to small. In AlexNet, large kernels (e.g., 11×11, 5×5) are used together with 3×3 kernels. Subsequently, VGG (Simonyan & Zisserman, 2015) suggests that a large kernel could be substituted by several 3×3 kernels utilizing fewer parameters. Then, the smallest 1×1 kernels are proposed for dimension reduction and efficient low dimensional embedding (Szegedy et al., 2015). Recently, due to the emergence of NAS, large kernels (e.g., 5×5, 7×7) have been reemerged and attracted the researchers' attention, and become one of the standard units for the searched CNNs. ProxylessNAS (Cai et al., 2019) argues that large kernels are beneficial for CNNs to preserve more information for the downsampling.

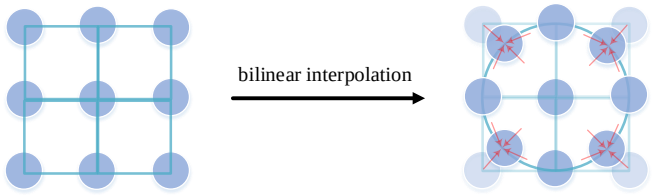
CNN Architecture Design. Since AlexNet achieved fundamental progress in the ILSVRC-2012 image classification competition (Krizhevsky et al., 2012), a number of outstanding manual CNN structures emerged, including VGG (Simonyan & Zisserman, 2015), Inception (Szegedy et al., 2015), ResNet (He et al., 2016), DenseNet (Huang et al., 2017), *etc*. However, designing the neural architecture heavily relies on researchers' prior knowledge, but existing prior knowledge and inherent mode of thinking are likely to limit the discovery of new neural architectures to a certain extent. As a result, neural architecture search (NAS) was developed to search good CNN structures automatically.

NAS-RL (Zoph & Le, 2017) and MetaQNN (Baker et al., 2017) using reinforcement learning (RL) are considered pioneers in the field of NAS. Subsequently, evolution-based algorithms use an evolving process towards better performance to search for novel neural architectures (Xie & Yuille, 2017; Real et al., 2017; Liu et al., 2018b; Elsken et al., 2019). To address the issue of high computational demand and time cost in the search scenario, the one-shot method constructs a super-net (Brock et al., 2018; Bender et al., 2018), which is trained once in search and then deemed as a performance estimator. Some studies sample a single path (Guo et al., 2019; Li & Talwalkar, 2019; You et al., 2020) in a chain-based search space (Hu et al., 2020; Cai et al., 2020; Mei et al., 2020; Yu et al., 2020a) to train the super-net. Another line of gradient-based methods (Liu et al., 2019; Chen et al., 2019; Chu et al., 2020; Xu et al., 2020; Chen & Hsieh, 2020; Yang et al., 2021; Chu et al., 2021) employs the gradient optimization method to perform differentiable joint optimization between the architecture parameters and the super-net weights in a cell-based space efficiently. Some gradient-based methods have reduced the search time significantly to about 0.1 GPU-days (Xu et al., 2020; Yang et al., 2021).

The search space of all the above works contains convolutions with large kernels that extensively exist in the final searched architectures. However, all the large kernels in these works directly inherit the square shape of the standard 3×3 kernel. Moreover, all the above works only employ operations that are popular in manual architectures to their operation space. Our work proposes to use convolutions of large circular kernels, which are more distinctive to the counterpart square kernels, to enrich the operation space for automatic neural architecture search. In turn, NAS may enable the convolutions of large circular kernels to be located in proper position to outperform the standard operations. As far as we know, this work is the first that involves the unpopular variants in the search space of NAS.

3 Circular Kernels for Convolution

This section introduces the circular kernel that evenly samples pixels on concentric circles to form circular receptive fields. We adopt bilinear interpolation for the approximation and re-parameterize the weight matrix by the corresponding transformation matrix to replace the receptive field offsets,



(a) Receptive field of a square kernel (b) Receptive field of a circular kernel

Figure 3: Approximation of a 3×3 circular kernel on a 3×3 square kernel

thus the training takes an approximately equivalent amount of calculation compared to the standard square convolution. We then provide preliminary analysis on the transformation matrix during training. In the end, we show how to incorporate large circular kernels into the NAS methods.

3.1 Circular Kernel versus Square Kernel

Without loss of generality, we take the 3×3 kernel as an example. The receptive field $\mathbb{S}$ of a 3×3 standard square kernel with dilation 1, as shown in Fig. 3 (a), can be presented as:

$$\begin{aligned}\mathbb{S} =\{&(-1,\ 1),(0,\ 1),(1,\ 1),(-1,\ 0),(0,\ 0),\\ &(1,\ 0),(-1,-1),(0,\ -1),(1,\ -1)\},\end{aligned} \tag{1}$$

where $\mathbb{S}$ denotes the set of offsets in the neighborhood considering the convolution conducted on the center pixel. By convolving an input feature map $\boldsymbol{I} \in \mathbb{R}^{H \times W}$ with a kernel $\boldsymbol{W} \in \mathbb{R}^{K \times K}$ of stride 1, we have an output feature map $\boldsymbol{O} \in \mathbb{R}^{H \times W}$, whose value at each coordinate $\boldsymbol{j}$ is:

$$\boldsymbol{O}_j = \sum_{s \in \mathbb{S}} \boldsymbol{W}_s \boldsymbol{I}_{j+s}. \tag{2}$$

So we have $\boldsymbol{O} = \mathbf{W} \otimes \boldsymbol{I}$ where $\otimes$ indicates a typical 2D convolution operation used in CNNs.

In contrast, the receptive field of a 3×3 circular kernel can be presented as:

$$\begin{aligned}\mathbb{R} =\{&(-\tfrac{\sqrt{2}}{2},\tfrac{\sqrt{2}}{2}),(0,\ 1),(\tfrac{\sqrt{2}}{2},\tfrac{\sqrt{2}}{2}),(-1,\ 0),(0,\ 0),\\ &(1,\ 0),(-\tfrac{\sqrt{2}}{2},-\tfrac{\sqrt{2}}{2}),(0,\ -1),(\tfrac{\sqrt{2}}{2},-\tfrac{\sqrt{2}}{2})\}.\end{aligned} \tag{3}$$

As shown in Fig. 3 (b), we resample the input $\boldsymbol{I}$ with a group of offsets to each discrete kernel position $\boldsymbol{s}$, denoted as $\{\Delta \boldsymbol{r}\}$, to form the circular receptive field. The corresponding convolution becomes:

$$\boldsymbol{O}_j = \sum_{\boldsymbol{s} \in \mathbb{S}} \boldsymbol{W}_{\boldsymbol{s}} \boldsymbol{I}_{\boldsymbol{j}+\boldsymbol{s}+\Delta \boldsymbol{r}}. \tag{4}$$

In other words, the value of each entry is the sum of the element-wise products of the kernel weights and the corresponding pixel values in the circular receptive field. As the sampling positions of a circular kernel contains fractional positions, we employ bilinear interpolation to approximate the corresponding sampling values inside the circular receptive field:

$$\boldsymbol{I}_{\boldsymbol{r}} = \sum_{\boldsymbol{s} \in \mathbb{S}} \mathcal{B}(\boldsymbol{s}, \boldsymbol{r}) \boldsymbol{I}_{\boldsymbol{s}}, \tag{5}$$

where $\boldsymbol{r}$ denotes a grid or fractional location in the circular receptive field, $\boldsymbol{s}$ enumerates all the grid locations in the corresponding square receptive filed, and $\mathcal{B}(\cdot, \cdot)$ is a two dimensional bilinear interpolation kernel. $\mathcal{B}$ can be separated into two one-dimensional kernels as $\mathcal{B}(\boldsymbol{s}, \boldsymbol{r}) = g(\boldsymbol{s}_x, \boldsymbol{r}_x) \cdot g(\boldsymbol{s}_y, \boldsymbol{r}_y)$, where $g(a, b) = max(0, 1 - |a - b|)$. So $\mathcal{B}(\boldsymbol{s}, \boldsymbol{r})$ is non-zero and in (0,1) only for the nearest four grids $\boldsymbol{s}$ in $\mathbb{S}$ around fractional location $\boldsymbol{r}$, and $\mathcal{B}(\boldsymbol{s}, \boldsymbol{r}) = 1$ only for the corresponding grid $\boldsymbol{s}$ in $\mathbb{S}$ for grid location $\boldsymbol{r}$.

3.2 Re-parameterization of the Weights

Implementing the convolution of a circular kernel that can operate efficiently is not trivial. Considering when building the convolution of a circular kernel, as the offsets of the sampling points in a circular

receptive field relative to a square receptive field are fixed, we extract the transformation matrix $\boldsymbol{B}$ of the whole receptive field by arranging $\mathcal{B}$ of one pixel $\boldsymbol{r}$ in Equation 5.

Let $\hat{\boldsymbol{I}}_{RF(\boldsymbol{j})} \in \mathbb{R}^{K^2 \times 1}$ and $\hat{\boldsymbol{W}} \in \mathbb{R}^{K^2 \times 1}$ represent the resized receptive field centered on the location j and the kernel, respectively. The standard convolution can be defined as $\boldsymbol{O}_j = \hat{\boldsymbol{W}}^\top \hat{\boldsymbol{I}}_{RF(\boldsymbol{j})}$. Then the convolution of circular kernel can be defined as:

$$\boldsymbol{O}_j = \hat{\boldsymbol{W}}^\top \left(\boldsymbol{B} \hat{\boldsymbol{I}}_{RF(\boldsymbol{j})} \right) = \left(\hat{\boldsymbol{W}}^\top \boldsymbol{B} \right) \hat{\boldsymbol{I}}_{RF(\boldsymbol{j})}, \tag{6}$$

where $\boldsymbol{B} \in \mathbb{R}^{K^2 \times K^2}$ is a fixed sparse coefficient matrix. Correspondingly, let $\boldsymbol{I} \in \mathbb{R}^{H \times W}$, $\boldsymbol{O} \in \mathbb{R}^{H \times W}$ and $\boldsymbol{W} \in \mathbb{R}^{K \times K}$ respectively represent the input feature map, output feature map and the kernel, the convolution of a circular kernel could be briefly defined as:

$$\boldsymbol{O} = \boldsymbol{W} \otimes (\boldsymbol{B} \star \boldsymbol{I}) = (\boldsymbol{W} \star \boldsymbol{B}) \otimes \boldsymbol{I}, \tag{7}$$

where $\boldsymbol{B} \star \boldsymbol{I}$ is to change the square receptive field to circular receptive field.

In this way, we could apply an operation on the kernel weights once to have $\boldsymbol{W} \star \boldsymbol{B}$ before the kernel scans the input feature map. Consequently, we do not need to calculate the offsets for each convolution as deformable methods do when the kernel scans the input feature map step by step (Jeon & Kim, 2017; Dai et al., 2017; Zhu et al., 2019; Gao et al., 2020). While calculating the receptive field offsets for each convolution is very time-consuming, the computational cost of operations on kernels is negligible compared to the gradient descent optimization.

3.3 Analysis on Transformation Matrix

This subsection briefly concludes the analysis of the actual effect of the transformation matrix. For a circular kernel, let $\Delta \boldsymbol{W} = \boldsymbol{W}^{t+1} - \boldsymbol{W}^t$. The squared value of a change on the output $\Delta \boldsymbol{O} = \boldsymbol{O}^{t+1} - \boldsymbol{O}^t$ can be calculated as $\|\Delta \boldsymbol{O}\|^2 = (\boldsymbol{B} \star \boldsymbol{I})^\top \otimes \Delta \boldsymbol{W}^\top \Delta \boldsymbol{W} \otimes (\boldsymbol{B} \star \boldsymbol{I})$, which can be transferred to $\|\Delta \boldsymbol{O}\|^2 = \boldsymbol{I}^\top \otimes \left(\boldsymbol{B}^\top \star \Delta \boldsymbol{W}^\top \Delta \boldsymbol{W} \star \boldsymbol{B} \right) \otimes \boldsymbol{I}$. In contrast, $\Delta \widetilde{\boldsymbol{O}}$ of the traditional convolutional layers is determined by $\Delta \boldsymbol{W}^\top \Delta \boldsymbol{W}$ and $\boldsymbol{I}$. Hence, we can conclude that the transformation matrix $\boldsymbol{B}$ affects the optimization paths of gradient descent. For detailed analysis, see the Supplementary. We also empirically demonstrate this claim in Section 5.2.

3.4 NAS with Large Circular Kernels

Theoretically, we could expand the operation space of any NAS method with large circular kernels. As the one-shot methods yield significant advantage in the time cost over the reinforcement learning or evolutionary-based NAS methods, and the typical one-shot methods of gradient-based ones (Liu et al., 2019; Chen et al., 2019; Xu et al., 2020; Yang et al., 2021) enable us to discover more complex connecting patterns, we adopt them as the baselines for incorporating the large circular kernels.

The search space for typical gradient-based methods is made up of cell-based microstructure repeats. Each cell can be viewed as a directed acyclic graph with N nodes and E edges, where each node x^i represents a latent representation (*e.g.*, a feature map), and each edge is associated with operations $o(\cdot)$ (*e.g.*, *identity connection*, *sep_conv_3x3*) in the operation space $\mathcal{O}$. Within a cell, the goal is to choose one operation from $\mathcal{O}$ to connect each pair of nodes.

Let a pair of nodes be (i, j), where $1 \leqslant i < j \leqslant N$, the core idea of typical gradient-based methods is to formulate the information propagated from i to j as a weighted sum over $|\mathcal{O}|$ operations as the mixed operation:

$$\bar{o}^{(i,j)}(w, \mathbf{x}_i) = \sum_{o \in \mathcal{O}} \frac{\exp(\alpha_o^{(i,j)})}{\sum_{o' \in \mathcal{O}} \exp(\alpha_{o'}^{(i,j)})} o\left(w_o^{(i,j)}, \mathbf{x}_i \right), \tag{8}$$

where $\mathbf{x}_i$ is the output of the i-th node, $\alpha_o^{(i,j)}$ is a hyper-parameter for weighting operation $o(\mathbf{x}_i)$, and $w_o^{(i,j)}$ is the weight. The entire framework is then differentiable to both layer weights in operation $o(\cdot)$ and hyper-parameters $\alpha_o^{(i,j)}$ in an end-to-end fashion. After that, a discrete architecture can be obtained by replacing mixed operations with most likely operations at the end of the search.

The operation space $\mathcal{O}$ of typical gradient-based methods is: 3×3 and 5×5 separable convolutions, 3×3 and 5×5 dilated separable convolutions, 3×3 max pooling, 3×3 average pooling, identity,

Table 1: Comparison with state-of-the-art searched network architectures on CIFAR-10. ‡ denotes model trained with AutoAugment Cubuk et al. (2018)

Architecture	Test Err. (%)	Params (M)	Search Cost (GPU-days)	Search Method
DenseNet-BC Huang et al. (2017)	3.46	25.6	-	manual
NASNet-A + cutout Zoph et al. (2018)	2.65	3.3	1800	RL
AmoebaNet-A + cutout Real et al. (2019)	3.34±0.06	3.2	3150	evolution
AmoebaNet-B + cutout Real et al. (2019)	2.55±0.05	2.8	3150	evolution
P-DARTS + cutout Chen et al. (2019)	2.50	3.4	0.3	gradient-based
R-DARTS(L2) + cutout Bender et al. (2018)	2.95±0.21	-	1.6	gradient-based
S-DARTS-ADV + cutout Chen & Hsieh (2020)	2.61±0.02	3.3	1.3	gradient-based
Fair DARTS + cutout Chu et al. (2020)	2.54±0.05	3.3	-	gradient-based
EnTranNAS-DST + cutout Yang et al. (2021)	2.48±0.08	3.2	0.1	gradient-based
DARTS− + cutout Chu et al. (2021)	2.59±0.08	3.5	0.4	gradient-based
DARTS + cutout Liu et al. (2019)	2.76±0.09	3.3	0.4	gradient-based
PC-DARTS + cutout‡ Xu et al. (2020)	2.15±0.04	3.6	0.1	gradient-based
DARTS-Circle + cutout	2.62±0.08	3.9	0.4	gradient-based
PC-DARTS-Circle + cutout‡	2.02±0.05	3.5	0.1	gradient-based

and *zero*, which only considers convolutions or pooling that are popular in manually designed CNNs. Considering variants of convolutions may outperform the popular convolutions when they are located in the proper position, we add 5×5 circular separable convolutions and 5×5 circular dilated separable convolutions to the operation space, which are constructed by replacing the square kernels of the separable or dilated convolution with circular kernels.

4 EXPERIMENTS

In this section, we empirically demonstrate the advantages of large circular kernels over large square kernels. Then we expand the search space of NAS with large circular kernels and apply the strategy described in several representative gradient-based NAS methods to search for new neural network architectures based on their ability of locating large circular kernels in proper position. Experimental results show that the searched architectures contain large circular kernels and outperform the original ones on both CIFAR-10 and ImageNet datasets. Detailed experimental setup can be found in the Supplementary.

4.1 THE ADVANTAGES OF LARGE CIRCULAR KERNELS

We have shown in Fig. 1 that a large circular kernel has a more round receptive field and is more distinguishable from the corresponding square kernel. We conjecture that the larger circular kernels would exhibit a more significant advantage over the square kernels if the circular kernels are helpful for deep learning tasks. To verify this hypothesis, we augment VGG (Simonyan & Zisserman, 2015), ResNet (He et al., 2016), WRNCifar (Zagoruyko & Komodakis, 2016), DenseNetCifar (Huang et al., 2017), and their circular kernel versions with larger kernel sizes and compare their performance on CIFAR-10 and CIFAR-100. For a fair comparison, we show the results of the original data without data augmentation. Results on CIFAR-10 and CIFAR-100 with standard data augmentation can be found in the Supplementary. We run the model 5 times for each dataset & model setting to reduce the variance.

As shown in Fig. 2, the performance of both the baselines and the corresponding circular kernel versions basically declines with the increment of the kernel size. Nevertheless, we see that the advantage of circular kernels over square kernels becomes more distinct for larger kernels. The average gain brought by circular kernels on four models is 1.1% for kernel size of 3, 2.0% for kernel size of 5, and 2.8% for kernel size of 7, indicating the superiority of large circular kernels.

4.2 SEARCHED MODELS WITH LARGE CIRCULAR KERNELS

Although the models using large circular kernels surpass the counterpart models using large square kernels, the large kernels are not superior to standard 3×3 kernels as a whole in the manually designed

Table 2: Comparison with state-of-the-art searched architectures on ImageNet (**mobile setting**). ‡: These architectures are searched on ImageNet directly, others are searched on CIFAR-10 or CIFAR-100 and transferred to ImageNet

Architecture	Test Err. (%)		Params	FLOPs	Search Cost	Search Method
	top-1	top-5	(M)	(M)	(GPU-days)	
ResNet50 He et al. (2016)	24.7	-	25.6	4100	-	manual
Inception-v1 Szegedy et al. (2015)	30.2	10.1	6.6	1448	-	manual
MobileNet Howard et al. (2017)	29.4	10.5	4.2	569	-	manual
ShuffleNet 2× (v2) Ma et al. (2018)	25.1	-	∼5	591	-	manual
NASNet-A Zoph et al. (2018)	26.0	8.4	5.3	564	1800	RL
AmoebaNet-C Real et al. (2019)	24.3	7.6	6.4	570	3150	evolution
FairNAS-A Chu et al. (2019)	24.7	-	4.6	388	12	evolution
P-DARTS Chen et al. (2019)	24.4	7.4	4.9	557	0.3	gradient-based
S-DARTS-ADV Chen & Hsieh (2020)	25.2	7.8	-	-	-	gradient-based
Fair DARTS Chu et al. (2020)‡	24.4	7.4	4.3	440	3.0	gradient-based
NSENet Ci et al. (2020)	24.5	-	4.6	330	-	gradient-based
DARTS Liu et al. (2019)	26.7	8.7	4.7	574	0.4	gradient-based
PC-DARTS (CIFAR-10) Xu et al. (2020)	25.1	7.8	5.3	586	0.1	gradient-based
PC-DARTS (ImageNet)‡ Xu et al. (2020)	24.2	7.3	5.3	597	3.2	gradient-based
DARTS-Circle	25.9	8.1	5.3	583	0.4	gradient-based
PC-DARTS-Circle (CIFAR-10)	24.9	7.7	5.0	571	0.1	gradient-based
PC-DARTS-Circle (ImageNet)‡	24.0	7.1	5.5	599	3.2	gradient-based
PC-DARTS-Circle-v2 (ImageNet)‡	23.7	7.0	5.7	639	3.2	gradient-based

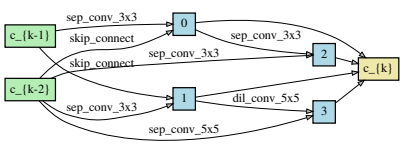
models. In this subsection, we incorporate the large circular kernels into the advanced gradient-based methods for neural architecture search so as to confirm the proper place of large circular kernels. DARTS (Liu et al., 2019) is the first NAS method based on joint gradient optimization, and PC-DARTS (Xu et al., 2020) is one of the best gradient-based methods. PC-DARTS enables a direct architecture search on ImageNet with only 3.8 GPU-days while most other NAS methods can only search on CIFAR and then evaluate on ImageNet. Hence, we incorporate the convolutions with large circular kernels to the operation space of DARTS and PC-DARTS. Denote our newly searched architectures as DARTS-Circle and PC-DARTS-Circle, respectively.

On CIFAR-10, the search and evaluation scenarios simply follow DARTS and PC-DARTS except for some necessary changes for a fair comparison. In the search scenario, the over-parameterized network is constructed by stacking 8 cells (6 normal cells and 2 reduction cells) for DARTS-Circle and PC-DARTS-Circle, and each cell consists of $N = 6$ nodes. In cell k, the first 2 nodes are input nodes, which are the outputs of cells $k-2$ and $k-1$, respectively. Each cell's output is the concatenation of all the intermediary nodes. In the evaluation stage, the network comprises 20 cells (18 normal cells and 2 reduction cells), and each type of cell shares the same architecture.

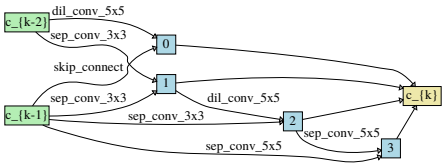
On ImageNet, following DARTS and PC-DARTS, the over-parameterized network starts with three convolution layers of stride 2 to reduce the input image resolution from 224×224 to 28×28. The cell architecture is the same with CIFAR-10. To reduce the search time, we randomly sample two subsets from the $1.3\mathrm{M}$ training set of ImageNet, with 10% and 2.5% images, respectively. In the evaluation stage, we apply the most popular *mobile setting* where the input image size is fixed to be 224×224, and the number of multi-add operations does not exceed $600\mathrm{M}$ (Liu et al., 2019; Xu et al., 2020; Chen et al., 2019).

The CIFAR-10 results for various convolutional architectures are presented in Table 1. Notably, both DARTS-Circle and PC-DARTS-Circle outperform the DARTS and PC-DARTS baselines with similar FLOPs, respectively. With AutoAugment (Cubuk et al., 2018), PC-DARTS has a surprising performance of 2.15% error rate. However, PC-DARTS-Circle can still boost the performance of PC-DARTS by $+0.13\%$ and achieves a state-of-the-art performance of 2.02% error rate with only 0.1 GPU-days.

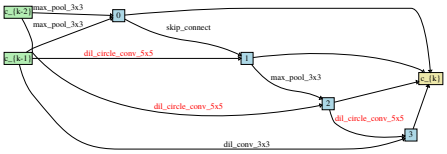
The comparisons on ImageNet are summarized in Table 2 and Fig. 4, illustrating that NAS methods have a better trade-off than the strong baseline of manual architecture. In NAS methods, DARTS-Circle achieves a top-1/5 error of $25.9\%/8.1\%$, considerably outperform-



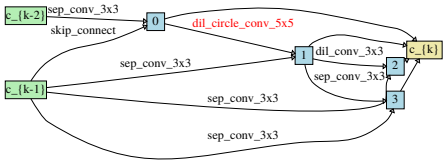
(a) The normal cell on CIFAR-10



(b) The normal cell on ImageNet



(c) The reduction cell on CIFAR-10



(d) The reduction cell on ImageNet

Figure 5: The searched normal cells and reduction cells of PC-DARTS-Circle on CIFAR-10 (*left*) and ImageNet (*right*). The circular kernels are marked in *red*

ing the results of 26.7%/8.7% reported by DARTS with similar FLOPs. Limited to the mobile setting, we reduce the 14 stacked cells used in PC-DARTS-Circle to 13 stacked cells. However, PC-DARTS-Circle can still achieve a state-of-the-art top-1/5 error of 24.0%/7.1%, slightly outperforming the results of 24.2%/7.3% reported by PC-DARTS. When PC-DARTS-Circle uses the same hyper-parameter of 14 stacked cells with PC-DARTS, denoted by PC-DARTS-Circle-v2, the top-1/5 error further reduces to 23.7%/7.0%, which is the best result of differentiable architecture search approaches under DARTS-based search space as far as we know. The main difference between PC-DARTS-Circle and PC-DARTS is that the former contains large circular kernels. So we can conclude that large circular kernels are excellent candidates for NAS. It is also worth noting that PC-DARTS-Circle outperforms NSENet, whose operation space contains 27 traditional operations without convolutions of circular kernels and is much more than the 9 operations employed in PC-DARTS-Circle.

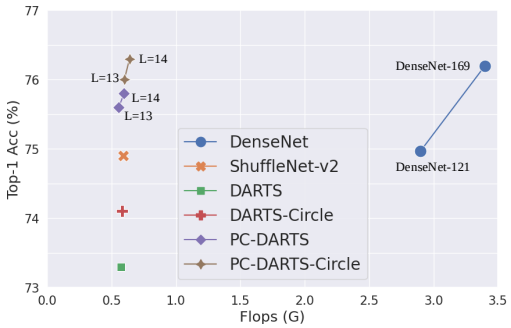
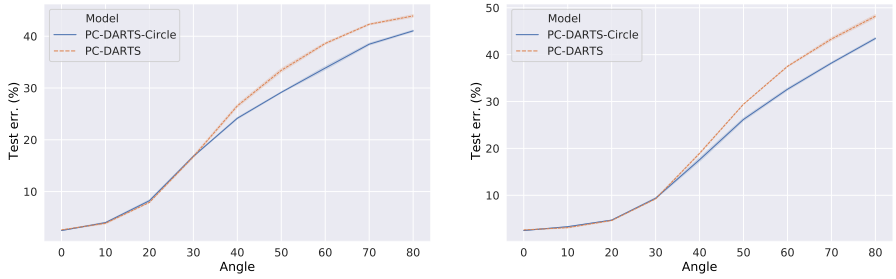


Figure 4: Comparison of top-1 accuracy on ImageNet with FLOPs (best viewed in color)

We visualize the searched normal cells and reduction cells of PC-DARTS-Circle on CIFAR-10 (*left*) and ImageNet (*right*) in Fig. 5. All other searched cells are shown in the Supplementary. Although large circular kernels only exist in a few layers and mainly exist in the reduction cells, they have a significant impact on the overall network because the receptive field is stacked as the layers go deeper.

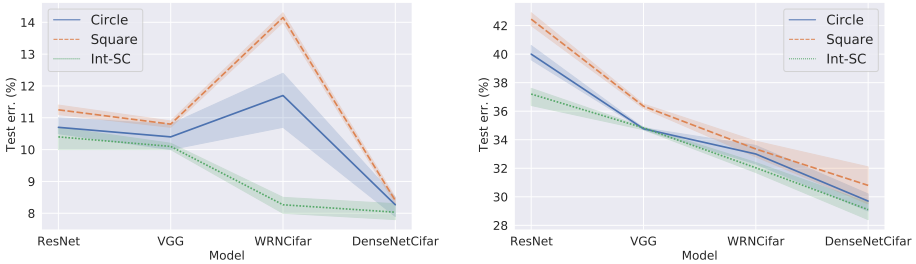
5 Further Analysis

For further analysis, we first show the better rotation invariance of large circular kernels compared to large square kernels by the evaluation on rotated or sheared images. Then, by constructing the integrated kernel, we reveal that the circular kernel has an optimization path different from the square kernel. In the Supplementary, we compare convolutions of large circular kernel with deformable convolutions (Jeon & Kim, 2017; Dai et al., 2017; Zhu et al., 2019) in NAS. The searched architecture containing convolutions of large circular kernels is with much less search and evaluation time cost and considerably outperforms the counterpart containing deformable convolutions.



(a) Rotation (b) Shear

Figure 6: Comparison of classification error on rotated images (*left*) or sheared images (*right*)



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

Figure 7: Test error (%) of the baselines with square kernels (*Square*) and the corresponding modified versions with circular kernels (*Circle*) or integrated kernels (*Int-SC*) on CIFAR-10 and CIFAR-100 without data augmentation (best viewed in color)

5.1 Rotation Invariance of Large Circular Kernels

To better understand the approximate rotation-invariant property of circular kernels, we investigate their robustness to rotated images or sheared images. Specifically, we compare the performance of PC-DARTS-Circle with PC-DARTS searched on CIFAR-10. They are both trained on the training set of CIFAR-10 with standard data augmentation. Fig. 6 illustrates the classification errors on the rotated or sheared images generated on the test set of CIFAR-10. The rotation or shear angle range takes the value in $\mathbb{D}_e = \{10, 20, 30, 40, 50, 60, 70, 80\}$. For each angle range $a \in \mathbb{D}_e$, the images are rotated or sheared with an angle uniformly sampled from $(-a, a)$, and we report the average classification error for three independent tests.

We can observe that the advantages of PC-DARTS-Circle steadily increase after $a > 30$, and reach the maximum at $a = 70$, which is roughly 4% for rotation and 5% for shear. The experiments not only justify the better rotation-invariant property of circular kernels, but also reveal that the rotation-invariant property of circular kernels in some layers is helpful to make the overall model more robust to the rotated images or sheared images.

5.2 Integrated Kernels and Optimal Path

In Section 3.3, we analyzed that the model with circular kernels has an optimal path different from that of the model with square kernels during the course of the gradient descent optimization. Here, we show their difference empirically by substituting all the 3×3 square kernels with the corresponding circular kernels or integrated kernels (explained in the followup paragraph) on WRNCifar (Zagoruyko & Komodakis, 2016) and DenseNetCifar (Huang et al., 2017).

We first introduce the integrated kernel in detail. Each integrated kernel has two candidate kernels containing a square kernel and a circular kernel, denoted by $\mathbb{D} = \{\mathbb{S}, \mathbb{R}\}$. At each iteration, we randomly select $\mathbb{D}_p \in \mathbb{D}$ according to a *binomial* distribution for each convolutional layer. Following Equation 4 in Section 3, we resample the input $\boldsymbol{I}$ with a group of offsets denoted by $\{\Delta \boldsymbol{d}\}$ that

corresponds to each discrete kernel position $\boldsymbol{s}$ to form the integrated receptive field. Then, the output feature map of the corresponding convolution is defined as $\boldsymbol{O}_j = \sum_{\boldsymbol{s}\in\mathbb{S}} \boldsymbol{W}_{\boldsymbol{s}} \boldsymbol{I}_{\boldsymbol{j}+\boldsymbol{s}+\Delta\boldsymbol{d}}$. The two types of kernels share the weight matrix but have distinct transformation matrices. During the training, the shared weight matrix is updated at each epoch, but the transformation matrices are randomly picked to determine the type of kernels of each layer at each iteration.

We compare the performance of the three versions of kernels on CIFAR-10 and CIFAR-100. The results are presented in Fig. 7. It is worth noting that the version with integrated kernels yields better results over the other two versions, even though the network architecture is manually designed based on square kernels. The superiority of the integrated kernels indicates that the switch between circular kernels and square kernels helps the model jump out of the local optima to perform better performance. Consequently, we can conclude that the circular kernel has an optimization path different from the square kernel for the gradient descent optimization.

6 CONCLUSION

In this work, we propose a new concept of circular kernel that could be an alternative option for the convolution operation in contemporary CNNs. The circular kernel exhibits approximately isotropic property and better rotation invariance because of the concentric and isotropic receptive field. We propose an simple yet efficient implementation of the convolution of circular kernels and reveal that the model with circular kernels has an optimization path different from that of the counterpart model with square kernels. Based on the increasing advantages of circular kernels over the counterpart square kernels with the increment of kernel size, we expand the operation space in several representative NAS methods with convolutions of large circular kernels because NAS enables large circular kernels to locate in proper position. We show that the searched architecture contains large circular kernels and outperforms the original architecture containing merely square kernels, and report state-of-the-art classification accuracy on benchmark datasets. Our work shows the potential of designing new shapes of convolutional kernels for CNNs, and bringing up the prospect of expanding the search space of NAS using variant of kernels that perform averagely as a whole in manual architecture but have extraordinary performance in the proper position in NAS.

REFERENCES

Bowen Baker, Otkrist Gupta, Nikhil Naik, and Ramesh Raskar. Designing neural network architectures using reinforcement learning. In *5th International Conference on Learning Representations, ICLR*, 2017.

Gabriel Bender, Pieter-Jan Kindermans, Barret Zoph, Vijay Vasudevan, and Quoc V. Le. Understanding and simplifying one-shot architecture search. In *Proceedings of the 35th International Conference on Machine Learning, ICML*, pp. 549–558, 2018.

Andrew Brock, Theodore Lim, James M. Ritchie, and Nick Weston. SMASH: one-shot model architecture search through hypernetworks. In *6th International Conference on Learning Representations, ICLR*, 2018.

Han Cai, Ligeng Zhu, and Song Han. ProxylessNAS: Direct neural architecture search on target task and hardware. In *7th International Conference on Learning Representations, ICLR*, 2019.

Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. Once-for-all: Train one network and specialize it for efficient deployment. In *8th International Conference on Learning Representations, ICLR*, 2020.

Xiangning Chen and Cho-Jui Hsieh. Stabilizing differentiable architecture search via perturbation-based regularization. In *Proceedings of the 37th International Conference on Machine Learning, ICML*, pp. 1554–1565, 2020.

Xin Chen, Lingxi Xie, Jun Wu, and Qi Tian. Progressive differentiable architecture search: Bridging the depth gap between search and evaluation. In *IEEE International Conference on Computer Vision, ICCV*, pp. 1294–1303, 2019.

François Chollet. Xception: Deep learning with depthwise separable convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 1800–1807, 2017.

Xiangxiang Chu, Bo Zhang, Ruijun Xu, and Jixiang Li. FairNAS: Rethinking evaluation fairness of weight sharing neural architecture search. *arXiv preprint arXiv:1907.01845*, 2019.

Xiangxiang Chu, Tianbao Zhou, Bo Zhang, and Jixiang Li. Fair DARTS: Eliminating unfair advantages in differentiable architecture search. In *European conference on computer vision, ECCV*, pp. 465–480, 2020.

Xiangxiang Chu, Xiaoxing Wang, Bo Zhang, Shun Lu, Xiaolin Wei, and Junchi Yan. DARTS-: robustly stepping out of performance collapse without indicators. In *9th International Conference on Learning Representations, ICLR*, 2021.

Yuanzheng Ci, Chen Lin, Ming Sun, Boyu Chen, Hongwen Zhang, and Wanli Ouyang. Evolving search space for neural architecture search. *arXiv preprint arXiv:2011.10904*, 2020.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *IEEE International Conference on Computer Vision, ICCV*, pp. 764–773, 2017.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 248–255, 2009.

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Xiaohan Ding, Yuchen Guo, Guiguang Ding, and Jungong Han. ACNet: Strengthening the kernel skeletons for powerful CNN via asymmetric convolution blocks. In *IEEE International Conference on Computer Vision, ICCV*, pp. 1911–1920, 2019.

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Efficient multi-objective neural architecture search via lamarckian evolution. In *7th International Conference on Learning Representations, ICLR*, 2019.

Hang Gao, Xizhou Zhu, Stephen Lin, and Jifeng Dai. Deformable kernels: Adapting effective receptive fields for object deformation. In *8th International Conference on Learning Representations, ICLR*, 2020.

Zichao Guo, Xiangyu Zhang, Haoyuan Mu, Wen Heng, Zechun Liu, Yichen Wei, and Jian Sun. Single path one-shot neural architecture search with uniform sampling. *arXiv preprint arXiv:1904.00420*, 2019.

Song Han, Jeff Pool, John Tran, and William J Dally. Learning both weights and connections for efficient neural networks. *arXiv preprint arXiv:1506.02626*, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 770–778, 2016.

Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Shoukang Hu, Sirui Xie, Hehui Zheng, Chunxiao Liu, Jianping Shi, Xunying Liu, and Dahua Lin. DSNAS: direct neural architecture search without parameter retraining. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 12081–12089, 2020.

Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 2261–2269, 2017.

David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, pp. 106–154, 1962.

Yunho Jeon and Junmo Kim. Active Convolution: Learning the shape of convolution for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 1846–1854, 2017.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR*, 2015.

Alex Krizhevsky. Learning multiple layers of features from tiny images. *Tech Report*, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *26th Annual Conference on Neural Information Processing Systems, NeurIPS*, pp. 1106–1114, 2012.

Alexander Krull, Tim-Oliver Buchholz, and Florian Jug. Noise2Void - learning denoising from single noisy images. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 2129–2137, 2019.

Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals. In *5th International Conference on Learning Representations, ICLR*, 2017.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, pp. 2278–2324, 1998.

Chen-Yu Lee, Saining Xie, Patrick W. Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, AISTATS*, 2015.

Liam Li and Ameet Talwalkar. Random search and reproducibility for neural architecture search. In *Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI*, pp. 367–377, 2019.

Xiang Li, Wenhai Wang, Xiaolin Hu, and Jian Yang. Selective kernel networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 510–519, 2019.

Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In *Proceedings of the European Conference on Computer Vision, ECCV*, pp. 19–34, 2018a.

Hanxiao Liu, Karen Simonyan, Oriol Vinyals, Chrisantha Fernando, and Koray Kavukcuoglu. Hierarchical representations for efficient architecture search. In *6th International Conference on Learning Representations, ICLR*, 2018b.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: differentiable architecture search. In *7th International Conference on Learning Representations, ICLR*, 2019.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022, 2021.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *arXiv preprint arXiv:2201.03545*, 2022.

Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard S. Zemel. Understanding the effective receptive field in deep convolutional neural networks. In *Annual Conference on Neural Information Processing Systems, NeurIPS*, pp. 4898–4906, 2016.

Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. ShuffleNet V2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European Conference on Computer Vision, ECCV*, 2018.

Franck Mamalet and Christophe Garcia. Simplifying convnets for fast learning. In *International Conference on Artificial Neural Networks, ICANN*, pp. 58–65, 2012.

Jieru Mei, Yingwei Li, Xiaochen Lian, Xiaojie Jin, Linjie Yang, Alan L. Yuille, and Jianchao Yang. AtomNAS: Fine-grained end-to-end neural architecture search. In *8th International Conference on Learning Representations, ICLR*, 2020.

Jim Mutch and David G Lowe. Object class recognition and localization using sparse features with limited receptive fields. *International Journal of Computer Vision, IJCV*, pp. 45–57, 2008.

Niv Nayman, Asaf Noy, Tal Ridnik, Itamar Friedman, Rong Jin, and Lihi Zelnik-Manor. XNAS: neural architecture search with expert advice. In *Annual Conference on Neural Information Processing Systems, NeurIPS*, pp. 1975–1985, 2019.

Chao Peng, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun. Large Kernel Matters - Improve semantic segmentation by global convolutional network. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 1743–1751, 2017.

Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc V. Le, and Alexey Kurakin. Large-scale evolution of image classifiers. In *Proceedings of the 34th International Conference on Machine Learning, ICML*, pp. 2902–2911, 2017.

Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. Regularized evolution for image classifier architecture search. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI*, pp. 4780–4789, 2019.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Xiaojiang Chen, and Xin Wang. A comprehensive survey of neural architecture search: Challenges and solutions. *arXiv preprint arXiv:2006.02903*, 2020.

Eero P Simoncelli and Bruno A Olshausen. Natural image statistics and neural representation. *Annual review of neuroscience*, pp. 1193–1216, 2001.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR*, 2015.

Zhun Sun, Mete Ozay, and Takayuki Okatani. Design of kernels in convolutional neural networks for image classification. In *Proceedings of the European Conference on Computer Vision, ECCV*, pp. 51–66, 2016.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 1–9, 2015.

Lingxi Xie and Alan L. Yuille. Genetic CNN. In *IEEE International Conference on Computer Vision, ICCV*, pp. 1388–1397, 2017.

Yuhui Xu, Lingxi Xie, Xiaopeng Zhang, Xin Chen, Guo-Jun Qi, Qi Tian, and Hongkai Xiong. PC-DARTS: partial channel connections for memory-efficient architecture search. In *8th International Conference on Learning Representations, ICLR*, 2020.

Yibo Yang, Shan You, Hongyang Li, Fei Wang, Chen Qian, and Zhouchen Lin. Towards improving the consistency, efficiency, and flexibility of differentiable neural architecture search. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 6667–6676, 2021.

Shan You, Tao Huang, Mingmin Yang, Fei Wang, Chen Qian, and Changshui Zhang. GreedyNAS: Towards fast one-shot NAS with greedy supernet. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 1996–2005, 2020.

Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *4th International Conference on Learning Representations, ICLR*, 2016.

Jiahui Yu, Pengchong Jin, Hanxiao Liu, Gabriel Bender, Pieter-Jan Kindermans, Mingxing Tan, Thomas Huang, Xiaodan Song, Ruoming Pang, and Quoc Le. Bignas: Scaling up neural architecture search with big single-stage models. In *Proceedings of the European Conference on Computer Vision, ECCV*, 2020a.

Kaicheng Yu, Christian Sciuto, Martin Jaggi, Claudiu Musat, and Mathieu Salzmann. Evaluating the search phase of neural architecture search. In *ICLR*, 2020b.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *Proceedings of the British Machine Vision Conference 2016, BMVC*, 2016.

Xinbang Zhang, Zehao Huang, Naiyan Wang, Shiming Xiang, and Chunhong Pan. You only search once: Single shot neural architecture search via direct sparse optimization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2021.

Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable ConvNets V2: more deformable, better results. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 9308–9316, 2019.

Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. In *5th International Conference on Learning Representations, ICLR*, 2017.

Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. Learning transferable architectures for scalable image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 8697–8710, 2018.

A APPENDIX

The first part of this supplementary is a detailed analysis of the transformation matrix in relation to Section 3.3 of the main text. Then, using CIFAR-10 and CIFAR-100 with standard data augmentation, we show the advantages of large circular kernels over large square kernels, related to Section 4.1 of the main text. The benefits of big circular convolutions over deformable convolutions are next demonstrated, which is relevant to Section 5 of the main text. Finally, there are details on experiments and visualization.

A.1 ANALYSIS ON THE TRANSFORMATION MATRIX

This section provides a theoretical analysis of the actual effect of the transformation matrix. Based on Equation 7 in the main text, for a circular convolution, the output feature map $\boldsymbol{O} = \boldsymbol{W} \otimes (\boldsymbol{B} \star \boldsymbol{I}) = (\boldsymbol{W} \star \boldsymbol{B}) \otimes \boldsymbol{I}$. Then the squared value of a change on the output $\Delta \boldsymbol{O} = \boldsymbol{O}^{t+1} - \boldsymbol{O}^{t}$ can be calculated as:

$$\begin{aligned} \|\Delta \boldsymbol{O}\|^2 &= \left(\Delta \boldsymbol{W} \otimes (\boldsymbol{B} \star \boldsymbol{I})\right)^\top \left(\Delta \boldsymbol{W} \otimes (\boldsymbol{B} \star \boldsymbol{I})\right) \\ &= (\boldsymbol{B} \star \boldsymbol{I})^\top \otimes \Delta \boldsymbol{W}^\top \Delta \boldsymbol{W} \otimes (\boldsymbol{B} \star \boldsymbol{I}), \end{aligned} \tag{9}$$

where $\Delta \boldsymbol{W}$ is defined as $\boldsymbol{W}^{t+1} - \boldsymbol{W}^{t}$. Here the magnitude of $\Delta \boldsymbol{O}$ is determined by the interaction between $\Delta \boldsymbol{W}^\top \Delta \boldsymbol{W}$ and $\boldsymbol{B} \star \boldsymbol{I}$, while $\Delta \widetilde{\boldsymbol{O}}$ of the traditional convolutional layers is determined by $\Delta \boldsymbol{W}^\top \Delta \boldsymbol{W}$ and $\boldsymbol{I}$. So the transformation matrix $\boldsymbol{B}$ actually warps the receptive field in the input feature map $\boldsymbol{I}$. And Equation 9 can be transferred to:

$$\begin{aligned} \|\Delta \boldsymbol{O}\|^2 &= \left((\Delta \boldsymbol{W} \star \boldsymbol{B}) \otimes \boldsymbol{I}\right)^\top \left((\Delta \boldsymbol{W} \star \boldsymbol{B}) \otimes \boldsymbol{I}\right) \\ &= \boldsymbol{I}^\top \otimes \left(\boldsymbol{B}^\top \star \Delta \boldsymbol{W}^\top \Delta \boldsymbol{W} \star \boldsymbol{B}\right) \otimes \boldsymbol{I}. \end{aligned} \tag{10}$$

Here the magnitude of $\Delta \boldsymbol{O}$ is determined by $\boldsymbol{B}^\top \star \Delta \boldsymbol{W}^\top \Delta \boldsymbol{W} \star \boldsymbol{B}$ and $\boldsymbol{I}$, while $\Delta \widetilde{\boldsymbol{O}}$ of traditional convolutional layers is determined by $\Delta \boldsymbol{W}^\top \Delta \boldsymbol{W}$ and $\boldsymbol{I}$. So the transformation matrix $\boldsymbol{B}$ can also be regarded as warping the kernel space. From Equation 9 and Equation 10, we can conclude that the transformation matrix $\boldsymbol{B}$ affects the optimal paths of gradient descent in both receptive field and kernel space.

A.2 THE ADVANTAGES OF LARGE CIRCULAR KERNELS

We have shown that a large circular kernel has a more round receptive field and is more distinguishable from the corresponding square kernel. We conjecture that the larger circular kernels would exhibit more significant advantage over the square kernels if the circular kernels are helpful for deep learning tasks. To verify this hypothesis, we augment VGG (Simonyan & Zisserman, 2015), ResNet (He et al., 2016), WRNCifar (Zagoruyko & Komodakis, 2016), DenseNetCifar (Huang et al., 2017), and their circular kernel versions with larger kernel sizes and compare their performance on CIFAR-10 and CIFAR-100. For a fair comparison, we show the results of the original data without data augmentation in the main text. Here, we show the results on CIFAR-10 and CIFAR-100 with standard data augmentation.

As shown in Table 3, the performance of both the baselines and the corresponding circular kernel versions basically declines with the increment of kernel size, because the original neural network architecture is designed and hence optimized on the 3×3 square kernel. Nevertheless, we see that the advantage of circular kernels over square kernels becomes more distinct for larger kernels, indicating the superiority of large circular kernels.

A.3 COMPARISON WITH DEFORMABLE CONVOLUTION

A natural idea of improving the operation space of NAS is to expand the search space with deformable convolutions (Jeon & Kim, 2017; Dai et al., 2017; Zhu et al., 2019), as deformable convolutions are flexible and work well in manual architectures but are never considered in existing NAS methods. However, because NAS is a complicated bilevel optimization problem even on the standard convolutions (Liu et al., 2019), convolutional adaptation methods that require additional parameters are unlikely to be applicable to NAS due to the excessive optimization and considerable computation overhead. We show the performance of searched architecture containing deformable convolutions

Table 3: Test error (%) of the baselines with square kernels (Square) and the corresponding circular kernel (Circle) versions in kernel size $k \in \{3, 5, 7\}$ on CIFAR-10 and CIFAR-100. With the increment of kernel size, the advantage of circular kernel over square kernel becomes more distinct

Model	**CIFAR-10**			**CIFAR-100**		
	Square	Circle	Test Err.↓	Square	Circle	Test Err.↓
VGG (3×3)	5.91 ± 0.04	5.81 ± 0.21	**0.10**	25.19 ± 0.12	25.10 ± 0.10	**0.09**
VGG (5×5)	6.96 ± 0.21	6.72 ± 0.11	**0.24**	28.08 ± 0.29	28.02 ± 0.06	**0.06**
VGG (7×7)	7.77 ± 0.06	7.62 ± 0.09	**0.15**	30.59 ± 0.29	29.99 ± 0.32	**0.60**
ResNet (3×3)	5.75 ± 0.03	5.74 ± 0.06	**0.01**	27.47 ± 0.19	27.56 ± 0.09	**-0.09**
ResNet (5×5)	6.33 ± 0.07	6.13 ± 0.16	**0.20**	28.00 ± 0.41	27.87 ± 0.26	**0.13**
ResNet (7×7)	6.82 ± 0.35	6.62 ± 0.07	**0.20**	29.11 ± 0.25	28.27 ± 0.03	**0.84**
WRNCifar (3×3)	4.21 ± 0.06	4.25 ± 0.09	**-0.04**	20.59 ± 0.18	21.10 ± 0.32	**-0.51**
WRNCifar (5×5)	4.58 ± 0.14	4.39 ± 0.18	**0.19**	21.24 ± 0.14	21.16 ± 0.12	**0.08**
WRNCifar (7×7)	5.18 ± 0.16	4.87 ± 0.06	**0.31**	22.44 ± 0.15	21.91 ± 0.18	**0.53**
DenseNetCifar (3×3)	5.05 ± 0.11	5.20 ± 0.09	**-0.15**	22.76 ± 0.20	22.62 ± 0.17	**0.14**
DenseNetCifar (5×5)	5.19 ± 0.12	5.15 ± 0.11	**0.04**	23.31 ± 0.41	22.96 ± 0.16	**0.35**
DenseNetCifar (7×7)	5.47 ± 0.34	5.36 ± 0.03	**0.11**	23.64 ± 0.19	23.26 ± 0.10	**0.38**

on CIFAR-10 in 4. Compared to PC-DARTS-Circle, PC-DARTS-Deformable takes 7 times search cost and 3 times evaluation cost. Due to the time constraints, we only train PC-DARTS-Deformable for 100 epochs, as the complete training for 600 epochs requires 28.8 GPU-days, which are 18 times evaluation cost of PC-DARTS-Circle.

Table 4: Comparison with searched network architecture containing deformable convolutions (PC-DARTS-Deformable) on CIFAR-10

Architecture	**Test Err. (%)**	**Params (M)**	**Search Cost (GPU-days)**	**Evaluation Cost (GPU-days)**
PC-DARTS-Deformable + cutout	4.28	5.7	0.7	4.8
PC-DARTS-Circle + cutout	2.54	3.5	0.1	1.4

A.4 More Details on Experiments

A.4.1 Manually Designed Models on CIFAR Datasets

This subsection provides additional details for comparing circular kernels versus square kernels by augmenting VGG (Simonyan & Zisserman, 2015), ResNet (He et al., 2016), WRNCifar (Zagoruyko & Komodakis, 2016), DenseNetCifar (Huang et al., 2017), with larger kernel sizes on CIFAR-10 and CIFAR-100 datasets. The two CIFAR datasets (Krizhevsky, 2009) consist of colored natural images in 32×32 pixels. The training set and test set contain $50,000$ and $10,000$ images, respectively. We train the models for 200 epochs with batch size 128 using weight decay 5×10^{-4} and report the test error of the final epoch. No augmentation or standard data augmentation (He et al., 2016; Huang et al., 2017; Larsson et al., 2017; Lee et al., 2015) (padding to 40×40, random cropping, left-right flipping) is employed. And we utilize the Stochastic Gradient Descent (SGD) optimizer with the momentum of 0.9. The learning rate initiates from 0.1 and gradually decays to zero following a half-cosine-function-shaped schedule with a warm-up at the first five epochs.

A.4.2 Searched Models on CIFAR-10

For the search and evaluation of DARTS-Circle and PC-DARTS-Circle on CIFAR-10, we follow the setup in DARTS and PC-DARTS.

In the search scenario, we train the network for 50 epochs. The 50K training set of CIFAR-10 is split into two equal-sized subsets, with one subset used for training the network weights and the other used to search the architecture hyper-parameters. The network weights are optimized by momentum SGD, with a learning rate annealed down to zero following a cosine schedule without restart, a momentum of 0.9, and a weight decay of 3×10^{-4}. For the architecture hyper-parameters, we employ an Adam

optimizer (Kingma & Ba, 2015), with a fixed learning rate of 6×10^{-4}, a momentum of $(0.5, 0.999)$, a weight decay of 10^{-3}, and initial number of channels 16. Following DARTS, DARTS-Circle has a batch size of 64 with an initial learning rate of 0.025. Following PC-DARTS, PC-DARTS-Circle has a batch size of 256 with an initial learning rate of 0.1. In DARTS-Circle, we found that almost all edges in the derived normal cell are connected with 5×5 circular separable convolutions under the configuration of DARTS. Considering the strategy of the edge selection in DARTS is not very reasonable as reported by (Chu et al., 2020), we use the *edge normalization* introduced in PC-DARTS to produce a more reasonable strategy of the edge selection.

In the evaluation stage, the models are trained from scratch for 600 epochs with a batch size of 96 and initial number of channels 36. We apply the SGD optimizer with an initial learning rate of 0.025 (annealed down to zero following a cosine schedule without restart), a momentum of 0.9, a weight decay of 3×10^{-4}, and a norm gradient clipping at 5. Cutout (DeVries & Taylor, 2017), as well as drop-path with a rate of 0.3 are also used for regularization.

A.4.3 Searched Models on ImageNet

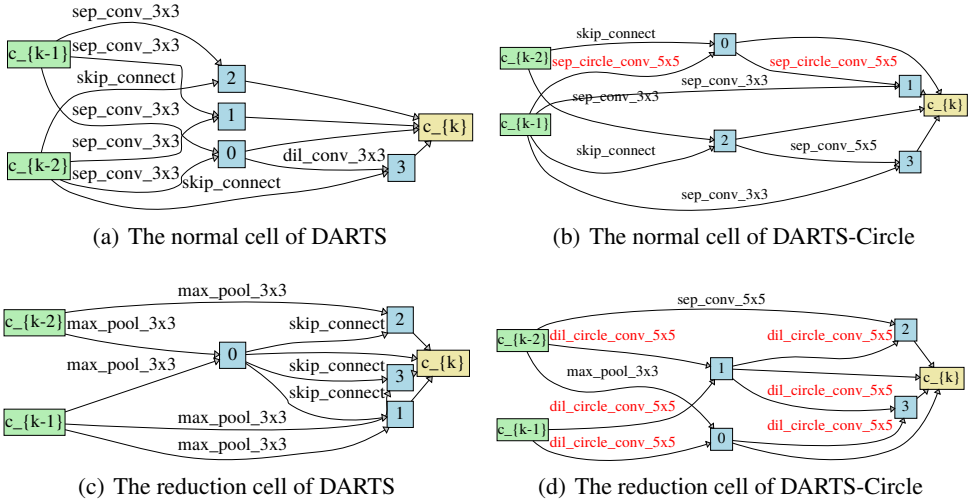
The ILSVRC 2012 classification dataset (Deng et al., 2009), ImageNet, consists of $1.3M$ training images and $50K$ validation images, all of which are high-resolution and roughly equally distributed over all the 1000 classes. The search and evaluation of DARTS-Circle and PC-DARTS-Circle on ImageNet follow DARTS (Liu et al., 2019) and PC-DARTS.

The search stage on ImageNet only exists in PC-DARTS-Circle. The model is trained for 50 epochs with a batch size of 1024. The architecture hyper-parameters are frozen during the first 35 epochs. For architecture hyper-parameters, we utilize the Adam optimizer (Kingma & Ba, 2015) with a fixed learning rate of 6×10^{-3}, a momentum of $(0.5, 0.999)$, and a weight decay of 10^{-3}. For the network weights, we utilize a momentum SGD with the initial learning rate of 0.5 (annealed down to zero following a cosine schedule without restart), a momentum of 0.9, and a weight decay of 3×10^{-5}.

In the evaluation stage, the models are trained from scratch for 250 epochs using a batch size of 512 and the initial channel number 48. We use the SGD optimizer with a momentum of 0.9, an initial learning rate of 0.25 (decayed down to zero linearly), and a weight decay of 3×10^{-5}. Additional enhancements are adopted, including label smoothing and an auxiliary loss tower during the training. The learning rate warm-up is applied for the first 5 epochs.

A.4.4 Visualization of the Searched Cells

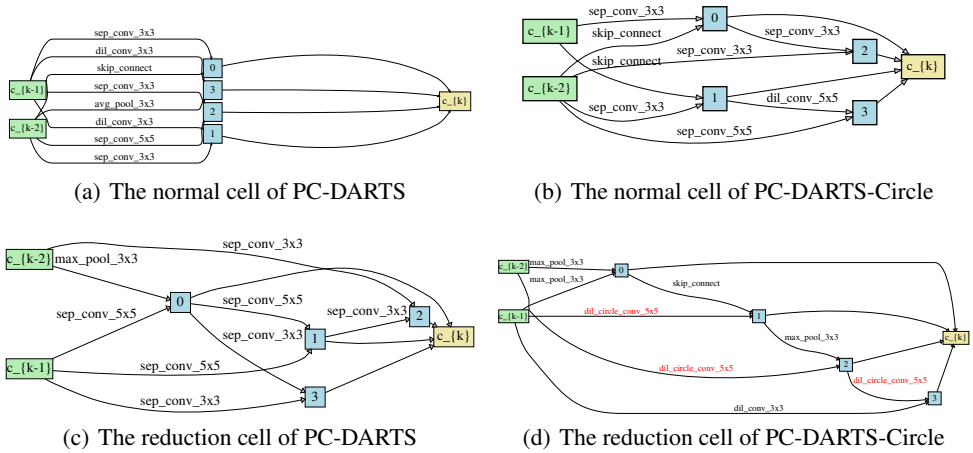
In this section, we visualize the searched normal cells and reduction cells for DARTS and DARTS-Circle on CIFAR-10, for PC-DARTS and PC-DARTS-Circle on CIFAR-10, and for PC-DARTS and PC-DARTS-Circle on ImageNet, in Fig. 8, Fig. 9, and Fig. 10, respectively. From the visualizations, we can observe that the normal cells of DARTS-Circle and PC-DARTS-Circle contain more large convolutions than those of the original versions. Additionally, convolutions of large circular kernels mainly exist in the reduction cells. According to DARTS (Liu et al., 2019), cells located at the $1/3$ and $2/3$ of the total depth of the network are reduction cells, in which all the operations adjacent to the input nodes are of stride two. We speculate that large circular kernels are significant when the size of feature maps change.



(a) The normal cell of DARTS (b) The normal cell of DARTS-Circle

(c) The reduction cell of DARTS (d) The reduction cell of DARTS-Circle

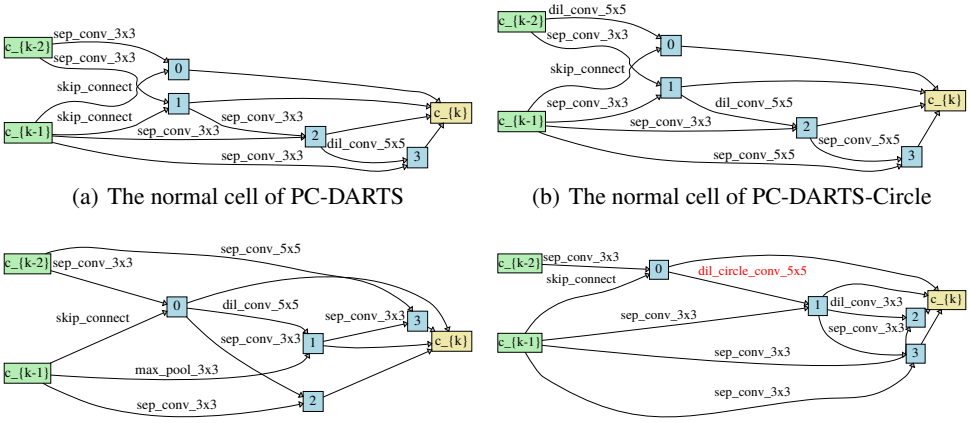
Figure 8: The searched normal and reduction cells of DARTS (*left*) and DARTS-Circle (*right*) on CIFAR-10. The large circular kernels are marked in *red*



(a) The normal cell of PC-DARTS (b) The normal cell of PC-DARTS-Circle

(c) The reduction cell of PC-DARTS (d) The reduction cell of PC-DARTS-Circle

Figure 9: The searched normal and reduction cells of PC-DARTS (*left*) and PC-DARTS-Circle (*right*) on CIFAR-10. The large circular kernel is marked in *red*



(a) The normal cell of PC-DARTS

(b) The normal cell of PC-DARTS-Circle

(c) The reduction cell of PC-DARTS

(d) The reduction cell of PC-DARTS-Circle

Figure 10: The searched normal and reduction cells of PC-DARTS (*left*) and PC-DARTS-Circle (*right*) on ImageNet. The large circular kernels are marked in *red*