INTEGRATING LARGE CIRCULAR KERNELS INTO CNNS THROUGH NEURAL ARCHITECTURE SEARCH

Anonymous authors
Paper under double-blind review

ABSTRACT

The square kernel is a standard unit for contemporary Convolutional Neural Networks (CNNs), as it fits well on the tensor computation for the convolution operation. However, the retinal ganglion cells in the biological visual system have approximately concentric receptive fields. Motivated by this observation, we propose using the circular kernel with a concentric and isotropic receptive field as an option for convolution operation. We first substitute the 3×3 square kernels with the corresponding circular kernels or our proposed integrated kernels in the typical ResNet architecture, and the modified models after training yield similar or even competitive performance. We then show the advantages of large circular kernels over the corresponding square kernels in that the difference and the improvement are more distinct. Hence, we speculate that large circular kernels would help find advanced neural network models by the Neural Architecture Search (NAS). To validate our hypothesis, we expand the operation space in several typical NAS methods with convolutions of large circular kernels. Experimental results show that the searched new neural network models contain large circular kernels and significantly outperform the original searched models. The additional empirical analysis also reveals that the large circular kernel help the model to be more robust to rotated or sheared images due to its rotation invariance.

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 the 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. A $K \times K$ circular kernel is defined as a kernel whose receptive field is concentrically and evenly sampled on K^2 pixels with the largest diameter of K. The circular kernels provide a number of advantages over the square kernels. First, the RFs and stacked RFs of circular kernels are more round and similar to the biological RFs. Second, the receptive field of a kernel is traditionally expected to be isotropic to 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 isotropic and roughly rotation-invariant, whereas a square kernel is symmetric only in a few orientations. Third, 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 weights at the four corners of large square kernels and stacked 3×3 square kernels are sparse. Compared to pruning these diluted parameters during the fine-tuning stage (Han et al., 2015), employing kernels with the same shape of effective RFs is probably a better option.

One of the cornerstones of the rationality of employing circular kernels is the isotropic feature of circles. However, a 3×3 circular kernel is not really in circular shape as it only samples nine pixels

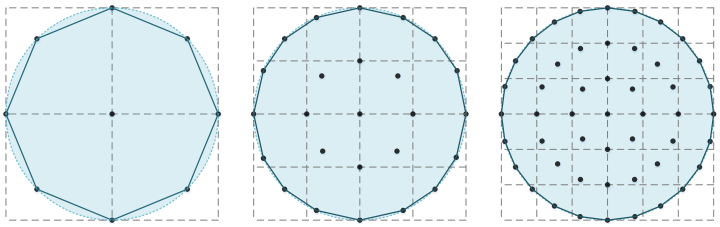


Figure 1: The receptive fields of circular kernels in size $k \in \{3, 5, 7\}$. All receptive fields are concentric and isotropic. A larger circular kernel has a more round receptive field.

with a similar arrangement to the square kernel. If we build the circular kernels in larger kernel size, as illustrated in Figure 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 reveal that the circular kernels exhibit 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 most powerful models generated by Neural Architecture Search (NAS) (Zoph et al., 2018; Liu et al., 2018a; Xu et al., 2020; Nayman et al., 2019) actually contain large square kernels, and many manually designed neural architectures also contain large square kernels (He et al., 2016a; Peng et al., 2017; Li et al., 2019). Large kernels have received insufficient attention despite their vast range of applications. Hence, we introduce large circular kernels to serve as excellent variants for existing large kernels.

The mainstream CNNs are manually developed and optimized on the 3×3 square kernel. So variants of kernels encounter significant resistance to outperform 3×3 square kernels. 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 operation space. In manual architectures, the network typically contains the same unit for all layers (e.g., 3×3 square kernels) because we do not know how to arrange them in different layers 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 right place as a part of the overall network. Then some special variants are likely to outperform the standard operations if it is placed in the right position. Consequently, although existing NAS methods have achieved superior performance, their operation space seems conservative that only contains popular operations used in manual architectures.

In this work, we propose to use the circular kernel with a concentric and isotropic receptive field as an option for the convolution operation. Our preliminary experiments show that simply substituting the square kernels in typical CNN architectures with circular kernels yields similar or even competitive classification performance. Then we show the increasing advantages of circular kernels over the square kernels as the size increases. We notice that these typical CNN architectures are manually designed and optimized based on the square kernels that are usually in size 3×3. It inspires us the possibility of designing new CNN architectures that are more favorable to circular kernels. To verify our hypothesis, we expand the operation space of several NAS methods with convolutions of large circular kernels as the difference is more distinct. Experimental results show that the searched architectures contain large circular kernels and outperform the original ones containing only square kernels for both CIFAR-10 and ImageNet datasets. Moreover, we demonstrate theoretically and experimentally that the model with circular kernels has different optimization path to that of the model with square kernels during the gradient descent process. And we also adopt additional experiments to reveal the rotation invariance of large circular kernels.

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 convolution kernel design and CNN architecture design, and 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 back to human's sight 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.

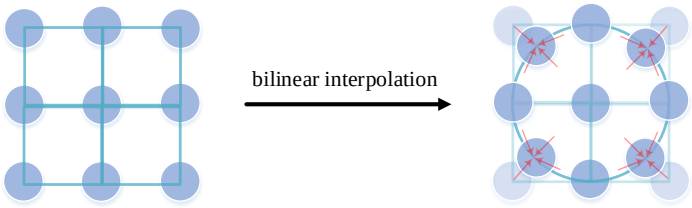
CNN Architecture Design. Since AlexNet achieved fundamental progress in the ILSVRC-2012 image classification competition (Krizhevsky et al., 2012), a number of outstanding manual CNN structures emerged, including VGG (Simonyan & Zisserman, 2015), Inception (Szegedy et al., 2015), ResNet (He et al., 2016a), DenseNet (Huang et al., 2017), *etc*. However, designing the neural architecture heavily relies on the 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, NAS was developed.

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., 2020) to train the super-net. Another line of DARTS-based methods (Liu et al., 2019; Chen et al., 2019; Xu et al., 2020; Yang 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 DARTS-based methods have reduced the search time significantly to about 0.1 GPU-days (Xu et al., 2020; Yang et al., 2021).

The operation 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 large circular kernels, that is more distinctive to the counterpart square kernels, to enrich the operation space for automatic neural architecture search.

3 Circular Kernels for Convolution

This section introduces the circular kernels with concentric and isotropic receptive fields for CNNs. We adopt bilinear interpolation for the approximation and re-parameterize the weight matrix by the corresponding transformation matrix to replace the receptive field offsets, thus the training takes an



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

Figure 2: Approximation of a 3×3 circular kernel on a 3×3 square kernel.

approximately equivalent amount of calculation compared to the standard square convolution. We then provide theoretical analysis on the effect of the transformation matrix during the training. In the end, we show how to incorporate large circular kernels into several advanced Neural Architecture Search (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 Figure 2 (a), can be presented as:

$$\mathbb{S} = \{(-1,\ 1), (0,\ 1\ ), (1,\ 1), (-1,\ 0), (0,\ 0), (1,\ 0), (-1, -1), (0,\ -1), (1,\ -1)\}, \quad (1)$$

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:

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

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:

$$\mathbb{R} = \{(-\tfrac{\sqrt{2}}{2}, \tfrac{\sqrt{2}}{2}), (0,\ 1), (\tfrac{\sqrt{2}}{2}, \tfrac{\sqrt{2}}{2}), (-1,\ 0), (0,\ 0), (1,\ 0), (-\tfrac{\sqrt{2}}{2}, -\tfrac{\sqrt{2}}{2}), (0,\ -1), (\tfrac{\sqrt{2}}{2}, -\tfrac{\sqrt{2}}{2})\}. \quad (3)$$

As shown in Figure 2 (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:

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

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 square receptive field:

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

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

When building 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 a pixel in Equation 5.

Let $\hat{\boldsymbol{I}}_{RF(\boldsymbol{j})} \in \mathbb{R}^{K^2 \times 1}$ and $\hat{\boldsymbol{W}} \in \mathbb{R}^{K^2 \times 1}$ respectively represent the resized receptive field centered on the location j and the kernel. The standard convolution can be defined as $\boldsymbol{O}_j = \hat{\boldsymbol{W}}^\top \hat{\boldsymbol{I}}_{RF(\boldsymbol{j})}$. Then the circular convolution 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 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 $\boldsymbol{B} \star \boldsymbol{I}$ represents 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 need not bother 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. Thus the training of a circular convolution takes an approximately equivalent amount of calculation compared to the standard square convolution.

3.3 Analysis on the Transformation Matrix

This subsection briefly concludes the theoretical analysis of the actual effect of the transformation matrix. For a circular kernel, the squared value of a change on the output $\Delta \boldsymbol{O} = \boldsymbol{O}^{t+1} - \boldsymbol{O}^t$ can be calculated as $\|\Delta \boldsymbol{O}\|^2 = (\boldsymbol{B} \star \boldsymbol{I})^\top \otimes \Delta \boldsymbol{W}^\top \Delta \boldsymbol{W} \otimes (\boldsymbol{B} \star \boldsymbol{I})$, $\Delta \boldsymbol{W} = \boldsymbol{W}^{t+1} - \boldsymbol{W}^t$, 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 optimal paths of gradient descent. For detailed analysis, see Appendix A. We also empirically demonstrate this claim in Section 4.1.

3.4 Neural Architecture Search with Large Circular Kernels

Theoretically, we could expand the operation space of any NAS method with 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 DARTS-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 circular kernels.

The search space for DARTS-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 an operation $o(\cdot)$ (*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. We add convolutions with large circular kernels to the operation space of the DARTS-based methods. The complete operation space $\mathcal{O}$ becomes: 3×3 and 5×5 separable convolutions, 3×3 and 5×5 dilated separable convolutions, 3×3 max pooling, 3×3 average pooling, identity, zero, 5×5 circular separable convolutions and 5×5 circular dilated separable convolutions. The last two operations are novel.

Let a pair of nodes be (i, j), where $0 \leqslant i < j \leqslant N - 1$, the core idea of DARTS-based methods is to formulate the information propagated from i to j as a weighted sum over $|\mathcal{O}|$ operations as the mixed operation:

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

where $\mathbf{x}_i$ is the output of the i-th node, and $\alpha_o^{(i,j)}$ is a hyper-parameter for weighting operation $o(\mathbf{x}_i)$. 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

Table 1: Test error (%) of ResNet-18 and ResNet-50 with standard square kernels and the modified versions with circular kernels or integrated kernels on the ImageNet dataset (lower error rate is better). *Square* and *Circle* respectively indicate the existing CNNs with square kernels and the counterpart CNNs with circular kernels. *Int-SC* indicates the integration of square and circular kernels.

Kernel	Epochs	ResNet-18			ResNet-50		
		Top-1 Err.	Top-5 Err.	FLOPs (G)	Top-1 Err.	Top-5 Err.	FLOPs (G)
$Square$	90	29.77	10.37	3.64	23.11	6.72	8.22
$Circle$	90	29.77	10.46	3.73	23.18	6.65	8.31
Int_SC	90	29.46	10.15	3.68	23.06	6.58	8.27

replacing mixed operations with the most likely operations at the end of the search. More details for the search process can be found in DARTS (Liu et al., 2019).

4 EXPERIMENTS

We empirically demonstrate the effectiveness of circular kernels on the image classification task using standard datasets. We first compare the 3×3 circular kernels and an integrated version of kernels with the 3×3 square kernels on ImageNet using manually designed typical neural architectures, ResNet (He et al., 2016a). Then we show the advantages of large circular kernels. In the main experiments, we expand the search space of NAS with large circular kernels and apply the strategy described in several representative DARTS-based methods to search for new neural network architectures. Experimental results show that the searched architectures contain large circular kernels and outperform the original ones that only contain square kernels for both CIFAR-10 and ImageNet datasets. An analysis on the robustness of circular kernels is provided in the end.

4.1 COMPARISON OF VARIOUS KERNELS ON RESNET

We know that the model with circular kernels has an optimal path different from that of the model with square kernels for 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 on two representative CNNs, ResNet-18 and ResNet-50 (He et al., 2016a). Detailed experimental setup can be found in Appendix B.1.

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, 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 $\boldsymbol{O}_j = \sum_{\boldsymbol{s} \in \mathbb{S}} \boldsymbol{W}_{\boldsymbol{s}} \boldsymbol{I}_{\boldsymbol{j}+\boldsymbol{s}+\Delta \boldsymbol{d}}$. If $\mathbb{D}_p = \mathbb{S}$, all the kernels in one layer are square kernels; and otherwise circular kernels. 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 at each iteration.

We compare the performance of the three versions of kernel on ImageNet. The results are presented in Table 1. The version with circular kernels yields similar results as compared with the version with square kernels, even though the network architecture is manually designed base on square kernels. It is worth noting that the version with integrated kernels yields better results over the other two versions. The superiority of integrated kernels indicates that circular kernel has an optimization path different from square kernel for the gradient descent optimization. Consequently, the switch between different optimal paths may help the model with integrated kernels jump out of the local optima to perform better performance.

4.2 THE ADVANTAGES OF LARGE CIRCULAR KERNELS

We have shown in Figure 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

Table 2: 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.↓
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

should 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 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. Detailed experimental setup for the comparison can be found in Appendix B.2.

As shown in Table 2, the performance of both the baselines and the corresponding circular kernel versions basically decays 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.

4.3 Searched Models with Large Circular Kernels

In this subsection, we incorporate the large circular kernels into the advanced DARTS-based methods for neural architecture search so as to evaluate the effectiveness 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 DARTS-based methods. PC-DARTS enables a direct architecture search on ImageNet with only 3.8 GPU-days while most of 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, respectively. Denote our newly searched architectures as DARTS-Circle and PC-DARTS-Circle, respectively.

On CIFAR-10, the search and evaluation scenario simply follow that of 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 4 cells (2 normal cells and 2 reduction cells) for DARTS-Circle and 8 cells (6 normal cells and 2 reduction cells) for 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 cells shares the same architecture. Detailed experimental setup can be found in Appendix B.3.

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 search scenario only exists in PC-DARTS. Then, 8 cells (6 normal cells and 2 reduction cells) are stacked beyond this point, and each cell consists of $N = 6$ nodes. To reduce the search time, we randomly sample two subsets from the 1.3M training set of ImageNet, with 10% and 2.5% images, respectively. The former is used for training the network weights and the latter is used for updating the hyper-parameters. In the evaluation stage, we apply the *mobile setting* where the input image size is fixed to 224×224, and the number of multi-add operations does not exceed 600M. Limited to the mobile setting, we reduce the 14 stacked cells used in DARTS and PC-DARTS to 13 stacked cells (11 normal cells and 2 reduction cells) for our models. See detailed experimental setup in Appendix B.4.

The CIFAR-10 results for convolutional architectures are presented in Table 3. Notably, both DARTS-Circle and PC-DARTS-Circle outperform the DARTS and PC-DARTS baselines respectively. The ImageNet results and comparison are summarized in Table 4. Notably, DARTS-Circle achieves a top-1/5 error of 25.9%/8.1%, significantly outperforming 26.7%/8.7% reported by DARTS. PC-

Table 3: Comparison with state-of-the-art searched network architectures on CIFAR-10.

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
ProxylessNAS + cutout (Cai et al., 2019)	2.08	-	4.0	gradient-based
P-DARTS + cutout (Chen et al., 2019)	2.50	3.4	0.3	gradient-based
BayesNAS + cutout (Zhou et al., 2019)	2.81±0.04	3.4	0.2	gradient-based
DARTS + cutout (Liu et al., 2019)	2.76±0.09	3.3	1	gradient-based
PC-DARTS + cutout (Xu et al., 2020)	2.57±0.07	3.6	0.1	gradient-based
DARTS-Circle + cutout	2.62±0.08	3.9	0.4	gradient-based
PC-DARTS-Circle + cutout	2.54±0.07	3.5	0.1	gradient-based

Table 4: Comparison with state-of-the-art searched architectures on ImageNet (mobile setting).

Architecture	Test Err. (%)		Params	$\times+$	Search Cost	Search Method
	top-1	top-5	(M)	(M)	(GPU-days)	
MobileNet (Howard et al., 2017)	29.4	10.5	4.2	569	-	manual
ShuffleNet 2× (v1) (Zhang et al., 2018)	26.4	10.2	∼5	524	-	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
ProxylessNAS (ImageNet)‡ (Cai et al., 2019)	24.9	7.5	7.1	465	8.3	gradient-based
P-DARTS (Chen et al., 2019)	24.4	7.4	4.9	557	0.3	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	1.0	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	3.8	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	2.4	gradient-based
PC-DARTS-Circle-v2 (ImageNet)‡	23.7	7.0	5.7	639	2.4	gradient-based

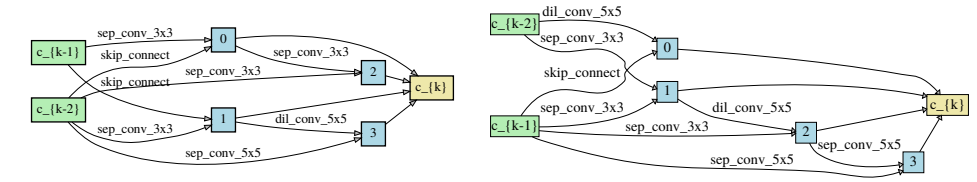
‡ These architectures are searched on ImageNet directly, others are searched on CIFAR-10 or CIFAR-100 and transferred to ImageNet.

DARTS-Circle achieves a state-of-the-art top-1/5 error of 24.0%/7.1% under the mobile setting, outperforming 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 reduces to 23.7%/7.0%. 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, much more than 10 operations employed in PC-DARTS-Circle.

We visualize the searched normal cells and reduction cells of PC-DARTS-Circle on CIFAR-10 (left-hand side) and ImageNet (right-hand side) in Figure 3. All other searched cells are shown in Appendix C. Although large circular kernels only exist in a few layers and mainly exist in the reduction cells, they can have a significant impact on the overall network because the receptive field is stacked as the layers go deeper.

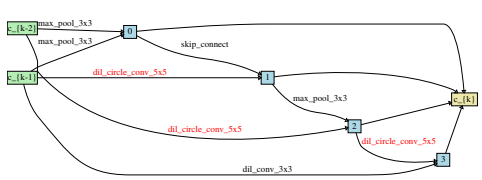
4.4 Analysis on the Robustness of Large Circular Kernels

To better understand the rotation-invariant property of circular kernels, we investigate their robustness to rotated 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

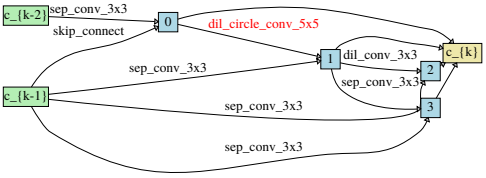


(a) The normal cell found on CIFAR-10

(c) The normal cell found on ImageNet



(b) The reduction cell found on CIFAR-10



(d) The reduction cell found on ImageNet

Figure 3: The searched normal cells and reduction cells of PC-DARTS-Circle on CIFAR-10 (left-hand side) and ImageNet (right-hand side). The circular kernels are marked in red.

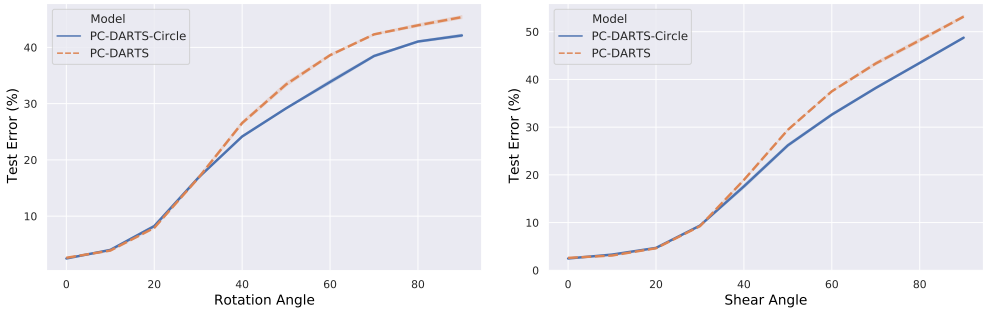


Figure 4: Comparison of classification error on rotated or sheared images.

standard data augmentation. Figure 4 illustrates the classification errors on the rotated or sheared images generated on the test set of CIFAR-10. The rotation or shear angle takes the value in $\mathbb{D}_e = \{10, 20, 30, 40, 50, 60, 70, 80, 90\}$. For each $angle = a$, 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 $angle > 30$, and reach the maximum at $angle = 70$, 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 rotated or sheared images.

5 Conclusion

This work proposes using the circular kernel with concentric and isotropic receptive field as an option for convolution operation, based on its rotation invariance and similarity to the biological RFs. We demonstrate theoretically and experimentally that a model with circular kernels has an optimization path different from that of the counterpart model with square kernels during the gradient descent process. Then we show the advantages of circular kernels over the square kernels as the size increases. By expanding the operation space in several representative NAS methods with large circular convolutions, 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 in particular on ImageNet. Further empirical evidence shows that large circular kernels are more robust to rotated or sheared images. In future work, it is possible to expand the operation space of NAS methods with special convolutions or poolings, which may not perform well in manually designed models as a whole but are superior in the right position of the searched models.

REPRODUCIBILITY STATEMENT

In Appendix B, we provide a complete description of the data processing steps for the datasets used in the experiments and all the detailed experimental setup. Furthermore, we repeat all the experiments on CIFAR-10 and CIFAR-100 datasets three times to reduce the variance, including Table 2, Table 3, and Figure 4. In Appendix C, we visualize all the searched models by our methods. We also cite all the baselines in the reference. We promise to provide the complete source code for the final version.

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.

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.

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.

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, 2016a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Proceedings of the European Conference on Computer Vision, ECCV*, pp. 630–645, 2016b.

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, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *Proceedings of the European Conference on Computer Vision, ECCV*, pp. 646–661, 2016.

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.

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*, 2020.

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

Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 6848–6856, 2018.

Hongpeng Zhou, Minghao Yang, Jun Wang, and Wei Pan. Bayesnas: A bayesian approach for neural architecture search. In *Proceedings of the 36th International Conference on Machine Learning, ICML*, pp. 7603–7613, 2019.

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 Analysis on the Transformation Matrix

This section provides a theoretical analysis of the actual effect of the transformation matrix. For a circular kernel, the squared value of a change on the output $\Delta \boldsymbol{O} = \boldsymbol{O}^{t+1} - \boldsymbol{O}^{t}$ can be calculated as:

$$\|\Delta \boldsymbol{O}\|^2 = (\Delta \boldsymbol{W} \otimes (\boldsymbol{B} \star \boldsymbol{I}))^\top (\Delta \boldsymbol{W} \otimes (\boldsymbol{B} \star \boldsymbol{I})) = (\boldsymbol{B} \star \boldsymbol{I})^\top \otimes \Delta \boldsymbol{W}^\top \Delta \boldsymbol{W} \otimes (\boldsymbol{B} \star \boldsymbol{I}), \quad (9)$$

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:

$$\|\Delta \boldsymbol{O}\|^2 = ((\Delta \boldsymbol{W} \star \boldsymbol{B}) \otimes \boldsymbol{I})^\top ((\Delta \boldsymbol{W} \star \boldsymbol{B}) \otimes \boldsymbol{I}) = \boldsymbol{I}^\top \otimes \left(\boldsymbol{B}^\top \star \Delta \boldsymbol{W}^\top \Delta \boldsymbol{W} \star \boldsymbol{B}\right) \otimes \boldsymbol{I}. \quad (10)$$

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.

B More Experimental Details

B.1 Manually Designed 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 comparison of various kernels on ResNet-18 and ResNet-50 follows the common practice (He et al., 2016a; Huang et al., 2016; 2017; He et al., 2016b). We train the models for 90 epochs using weight decay 1×10^{-4} using the standard data augmentations with batch size 256. 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. We report both the top-1 and top-5 accuracy of typical CNN models and our corresponding circular or integrated versions on the validation set at the final epoch.

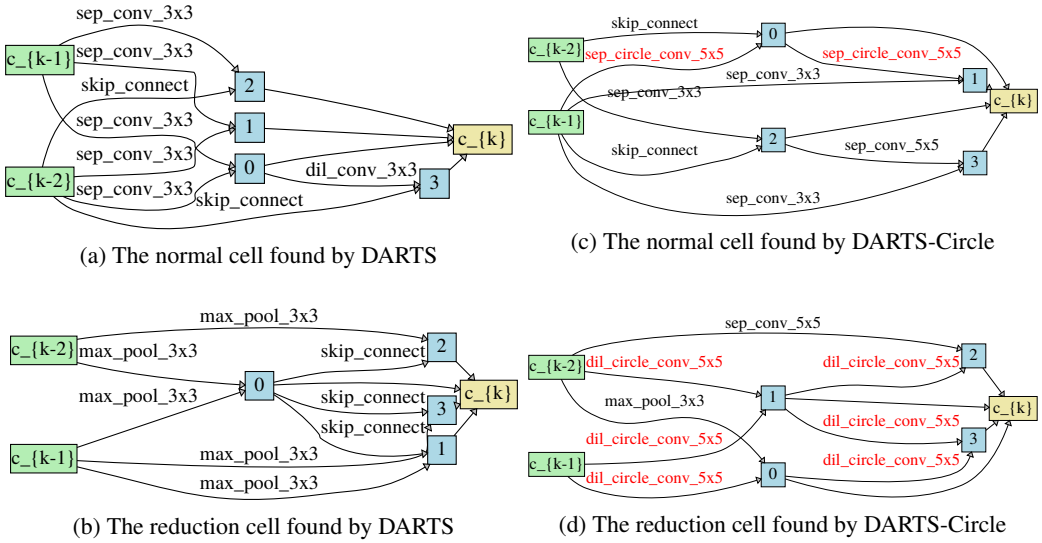
B.2 Manually Designed Models on CIFAR Datasets

This subsection provides additional details for the comparison of circular kernels versus square kernels by augmenting DenseNetCifar and WRNCifar 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 standard data augmentation (He et al., 2016a; Huang et al., 2017; Larsson et al., 2017; Lee et al., 2015) (padding to 40×40, random cropping, left-right flipping) and report the test error of the final epoch. 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. We evaluate the test error three times for each dataset & model setting to reduce the variance.

B.3 Searched Models on CIFAR-10

For the search and evaluation of DARTS-Circle and PC-DARTS-Circle on CIFAR-10, we basically 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



(a) The normal cell found by DARTS

(c) The normal cell found by DARTS-Circle

(b) The reduction cell found by DARTS

(d) The reduction cell found by DARTS-Circle

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

momentum of $(0.5, 0.999)$, and a weight decay of 10^{-3}. Following PC-DARTS, both DARTS-Circle and PC-DARTS-Circle have a batch size 256 with an initial learning rate 0.1. The initial number of channels is 9 for DARTS-Circle and 16 for PC-DARTS-Circle. 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. So 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.

B.4 Searched Models on ImageNet

The search and evaluation of DARTS-Circle and PC-DARTS-Circle on ImageNet basically follow DARTS (Liu et al., 2019) and PC-DARTS.

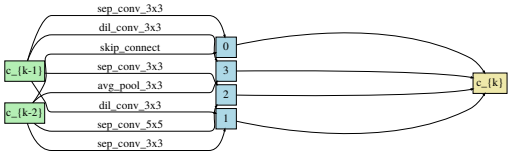
The search stage only exists in PC-DARTS-Circle, with a total of 50 epochs be trained, and the architecture hyper-parameters are frozen during the first 35 epochs. For architecture hyper-parameters, we utilize the Adam optimizer (Kingma & Ba, 2015) with a fixed learning rate of 6×10^{-3}, a momentum 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}. We employ three Tesla V100 GPUs for search with a total batch size of 512.

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.

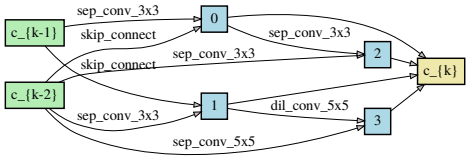
C Visualization of the Searched Cells

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

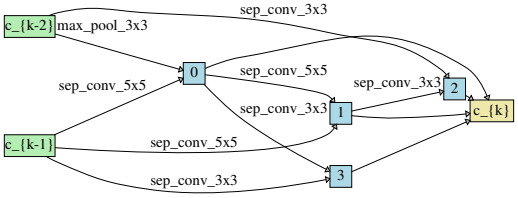
and PC-DARTS-Circle on ImageNet in Figures 5, 6 and 7, respectively. From the visualizations, we can observe that the normal cells of DARTS-Circle and PC-DARTS-Circle contain more large convolutions compared to those of the original versions. Additionally, large circular convolutions mainly exist in 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.



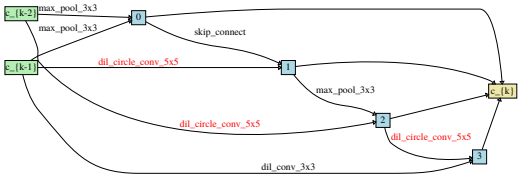
(a) The normal cell found by PC-DARTS



(c) The normal cell found by PC-DARTS-Circle

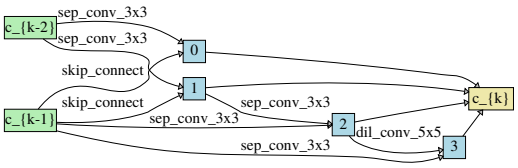


(b) The reduction cell found by PC-DARTS

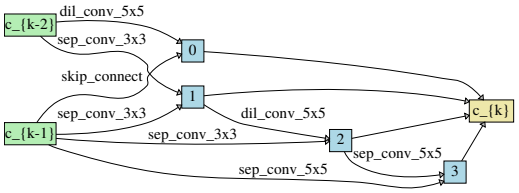


(d) The reduction cell found by PC-DARTS-Circle

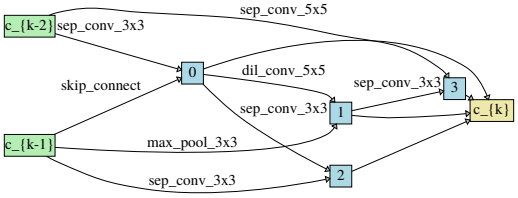
Figure 6: The searched normal and reduction cells of PC-DARTS (left-hand side) and PC-DARTS-Circle (right-hand side) on CIFAR-10. The large circular kernel is marked in red.



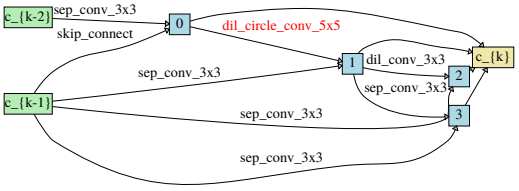
(a) The normal cell found by PC-DARTS



(c) The normal cell found by PC-DARTS-Circle



(b) The reduction cell found by PC-DARTS



(d) The reduction cell found by PC-DARTS-Circle

Figure 7: The searched normal and reduction cells of PC-DARTS (left-hand side) and PC-DARTS-Circle (right-hand side) on ImageNet. The large circular kernels are marked in red.