4-Connected Shift Residual Networks

Andrew Brown, Pascal Mettes, and Marcel Worring
University of Amsterdam
a.g.brown,p.s.m.mettes,m.worring@uva.nl

Abstract

The shift operation was recently introduced as an alternative to spatial convolutions. The operation moves subsets of activations horizontally and/or vertically. Spatial convolutions are then replaced with shift operations followed by point-wise convolutions, significantly reducing computational costs. In this work, we investigate how shifts should best be applied to high accuracy CNNs. We apply shifts of two different neighbourhood groups to ResNet on ImageNet: the originally introduced 8-connected (8C) neighbourhood shift and the less well studied 4-connected (4C) neighbourhood shift. We find that when replacing ResNet's spatial convolutions with shifts, both shift neighbourhoods give equal ImageNet accuracy, showing the sufficiency of small neighbourhoods for large images. Interestingly, when incorporating shifts to all point-wise convolutions in residual networks, 4-connected shifts outperform 8-connected shifts. Such a 4-connected shift setup gives the same accuracy as full residual networks while reducing the number of parameters and FLOPs by over 40%. We then highlight that without spatial convolutions, ResNet's downsampling/upsampling bottleneck channel structure is no longer needed. We show a new, 4C shift-based residual network, much shorter than the original ResNet yet with a higher accuracy for the same computational cost. This network is the highest accuracy shift-based network yet shown, demonstrating the potential of shifting in deep neural networks.

1. Introduction

In recent years, convolutional neural networks (CNNs) have radically improved the state-of-the-art in image classification accuracy. Yet this improvement has come at an exponentially increasing computational expense. The popular CNN ResNet [10] (2016) is approximately ten times the computational cost of earlier ImageNet challenge [25] winners such as AlexNet [18] (2012). The most accurate networks on ImageNet today, SENet154 [13] (2018) and NasNet [37] (2018), are approximately twice as expensive as ResNet.

This work focuses on optimising the computational footprint of high accuracy networks by replacing one of their costliest components in terms of parameters and FLOPs, the spatial convolution. We work with ResNet, as this popular network is still close to the most accurate networks today, and modify its architecture with the shift operation. The shift operation was recently introduced by Wu *et al* [32] and moves all elements within a given channel's image plane horizontally and/or vertically, with different (groups of) channels undertaking different moves. The operation is FLOP and parameter free, being theoretically equivalent to a re-referencing of the initial activations maps [11]. Spatial convolutions are replaced by a shift followed by a point-wise (1×1) convolution, itself equivalent to a simple matrix multiplication, and so require fewer parameters and FLOPs.

So far, several shift-based CNN architectures have been proposed [11, 16, 32] for the small-scale image datasets CIFAR10 and CIFAR100 [17]. Computationally constrained CNN architectures [11, 16, 32] have also been proposed for large images on ImageNet [25]. High accuracy shift network architectures, equalling or surpassing ResNet [10] on ImageNet, have not yet been explored. Here, we ask: for high accuracy networks such as ResNet, how should shifts be applied, and what discrete shift neighbourhood is best? The question of the spatial extent of neighbourhoods in visual recognition is a long-standing challenge, dating back to cellular arrays in image processing [6] and subsequently in image filtering [27]. For the spatial extent in rectangular arrays, two neighbourhoods are generally employed: the 8-connected (8C) neighbours (left, right, up and down + diagonals); and the 4-connected (4C) neighbours (left, right, up and down only). These neighbourhoods, illustrated in Fig. 1, are also known as the *Moore neighbourhood* and *von Neumann neighbourhood*, respectively [22]. Here we look at which neighbourhood to use in the high-accuracy deep learning setting.

The main focus of this work is two-fold. First, we aim to employ shifts on a full ResNet network (ResNet101) on a large-scale image dataset. This is with a view to optimising network architecture, either by maintaining accuracy while cutting computational cost, or by maintaining computational cost while improving accuracy. Second, we investigate which neighbourhood extent is sufficient for image recognition in such networks. The original shift replaces the 3×3 spatial convolutional kernel with 3×3 shifts. Implicitly, they opt for the Moore neighbourhood, but is such a full extent necessary?

In line with these focus points, we propose two extensions of the ResNet architecture. The first network adds multiple shifts to ResNet's residual blocks to reduce FLOPs and maintain accuracy. The second network focuses on accuracy for the same FLOPs. We highlight that, without the spatial convolutions, the 'bottleneck' in ResNet's channel structure is no longer needed. We construct a shorter network with a simpler channel structure, without down- and up-sampling, and show it gives superior performance on ImageNet. We then make the following contributions:

- We explore alternative neighbourhoods variants of the shift operation in ResNet on ImageNet. When directly replacing spatial convolutions, we find that shifting only to the 4-connected neighbours is sufficient for image recognition.
- We propose a multi-shift architecture, adding spatial information to the downsampling and up-sampling convolutions of ResNet. We find that performance is improved with this approach, but only for 4-connected shifts. This result highlights the importance of constraining neighbourhood extents when shifting on large networks. Our proposed multi-shift network then reduces ResNet's computational costs by 43% while maintaining accuracy.
- We propose a multi-shift-based ResNet variant without the 'bottleneck', which becomes possible when replacing spatial convolutions with shifts. The channel structure then becomes less complex, as the same number of channels is used throughout each residual block, and the network much shorter (35 layers) than the original (101 layers). We show a network with this design using 4-connected shifts which has approximately the same computational costs as ResNet101 and an accuracy increase of +0.8%. This is the highest accuracy shift-based network ever demonstrated on ImageNet.

2. Related Work

The shift operation was first introduced in [32]. Shifts translate activation maps horizontally and/or vertically to a neighbouring position. The shift operations of [32] consider a square 8-connected neighbourhood for image classification, with the shift-based CNN architectures demonstrated in [32] primarily optimised for the miniature image datasets CIFAR10/100 [17]. Computationally constrained networks for larger images, tested on ImageNet [25], were also shown. A higher accuracy network (ShiftResNet50) for ImageNet was also tested, but its architecture is unpublished. We estimate the FLOPs of this architecture and compare its results to ours in this work. We build on [32]; we focus on varying the shift neighbourhood and applying it to high accuracy networks for large images. We furthermore propose new residual architectures for shifts, resulting in competitive networks with low computational cost, outperforming other shift approaches, such as Wu et al. [32].

Other works have also investigated varying shifts operations for image classification. Closely related to this work is that of [11], who vary the (discrete) neighbourhood of shifts for miniature images. They then build a compact model for large images (accuracy 67.0%), though the FLOPs and parameters of this model were not shown. Comparatively, we focus on comparing shift neighbourhoods for the large image setting and for much larger, high accuracy networks (78.4%). We explore additional shift variants, discussing when they are appropriate, showing the architectural changes required to optimise large shift networks.

Active-shifts, introduced in [16], also relate to this work. By realising shifts as bi-linear interpolations of an input activation map, the horizontal and vertical motions of a shift can treated as real, trainable values. However, active-shifts require additional FLOPs to calculate these interpolations. Further, as activation map motion is non-integer, active shifts always require additional activation map copies in any implementation [11]. [16] also focus on optimising network architectures for miniature image datasets and for computationally constrained models on ImageNet, while we go beyond small datasets and compact networks for shifting.

Most recently, sparse-shifts were introduced in [31]. Sparse shifts attempt to learn discrete shift neighbourhoods by integer approximations of active-shifts. While [31] do consider constraining shift neighbourhoods through an L1 regularisation of shift magnitude, they ultimately find unconstrained shift-neighbourhoods to be optimal, in contrast to our results. We show the results of [31] in Fig. 2.

Shifts have also been applied in other contexts. These range from optimising shift-based CNNs for use with FPGAs [34] or systolic arrays [19] to re-purposing shifts for new tasks such as video recognition [20]. To our knowledge, no work has yet explored how shifts should be applied to a high accuracy network for image classification, or explored shift neighbourhoods in this setting.

In a broader sense, our work relates to methods to reduce network computational cost of larger CNNs. Examples of such methods are in network design (e.g. [26, 15]), tensor decomposition (e.g. [3, 4]), network pruning (e.g. [9, 8]) and student-teacher network training (e.g. [12, 24]). Shift operations in general and the shift-variants and CNN architectures we consider here are both complementary to and distinct from these approaches.

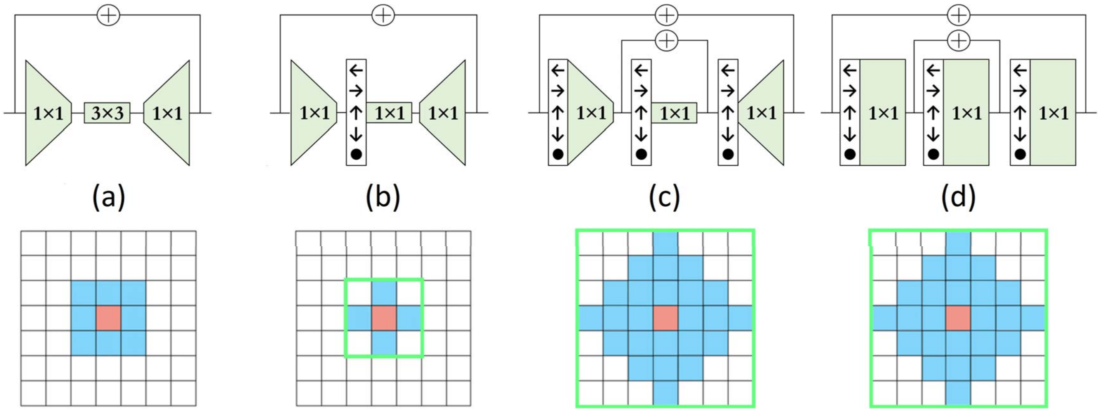

Figure 1. (Top) Original and proposed residual block designs. Green blocks are convolutions; the height of the blocks correspond to the number of channels. White boxes with arrows indicate shifts. Batch normalization and ReLU operations, applied after each convolution, are omitted for clarity. From left to right: (a) the down- and up- sampling bottleneck design of the original ResNet, (b) a single 4-connected shift residual network, with the bottleneck channel design (c) our proposed multi-shift block, with shifts applied before every 1×1 convolution, also with the bottleneck design, and (d) our simplified channel-flattened multi-shift residual block, which is enabled through the replacement of spatial convolutions. (Bottom) The theoretical receptive field extent of each residual block. Elements within the receptive field of each block are shown in blue, with the origin indicated in red. Green boxes indicate receptive field extents if 8-connected shifts are used instead of 4-connected shifts.

3. Method

3.1. The 4-connected shift operation

The shift operation, introduced in [32], moves all the elements of an activation map an integer number of elements along spatial directions. The set of allowed shift directions is the *shift neighbourhood*. Different (subsets of) channels are moved to different positions within this neighbourhood. In the original design, the shift neighbourhood matches the square neighbourhood of the equivalent square convolutional kernel of a spatial convolution (Fig. 1 a). Thus, to match a square kernel of spatial extent $D_k \times D_k$, there are $K = D_k^2$ neighbourhood positions. To perform the operation, an activation map of M channels is split into D_k^2 subgroups, each of a size $M//D_k^2$ channels, where '//' denotes integer division. Each of the K channel subgroups' activation maps is then moved in one of the K neighbourhood positions. In the case that division $M//D_k^2$ is not exact, the remaining ($M \bmod D_k^2$) channels are added to the origin (central element) subgroup and are, in practice, unmoved.

As stated in the introduction, here we ask: what shift neighbourhood is optimal? We compare the two neighbourhoods which have a long history of importance in image processing: the 8-connected (8C) Moore neighbourhood and the 4-connected (4C) von Neumann neighbourhood [22, 23, 30, 5]. Fig. 1 visually compares these neighbourhoods. Most modern CNN frameworks, such as TensorFlow [7] or Pytorch [21] only allow rectangular convolutional kernels in spatial convolution operations, and do not allow, for example, the cross shape of 4C neighbourhoods. Shift operations provide a new opportunity to study image neighbourhood connectivity, as they do not rely on these spatial convolution operations.

More formally, for two point-sets $\mathbf{X}$ and $\mathbf{Z}$, corresponding to the input and output activation maps of a shift, we define the neighbourhood function N from $\mathbf{X}$ to $\mathbf{Z}$ [22]:

$$N : \mathbf{X} \rightarrow 2^{\mathbf{Z}}, \tag{1}$$

such that for each point $\mathbf{x} \in \mathbf{X}$, it holds that $N(\mathbf{x}) \subset \mathbf{Z}$. The 8C neighbourhood and 4C neighbourhood functions are defined as:

$$\begin{aligned} N_{8C}(\mathbf{x}) =& \{\mathbf{y} : \mathbf{y} = (x_1 \pm a, x_2 \pm b) \,, \\ & a, b \in \{0, 1\}\}, \end{aligned} \tag{2}$$

$$\begin{aligned} N_{4C}(\mathbf{x}) =& \{\mathbf{y} : \mathbf{y} = (x_1 \pm a, x_2) \; or \; \mathbf{y} = (x_1, x_2 \pm b), \\ & a, b \in \{0, 1\}\}. \end{aligned} \tag{3}$$

Noting the results of [11], we are also interested if the origin element, $a = b = 0$, is strictly necessary. In residual networks, information about the origin element can be carried by the residual connection itself. It might then be natural to not also include origin element information in shift operations used in residual networks. We first define the origin or 'no-shift' neighbourhood as:

$$N_O(\mathbf{x}) = \{\mathbf{y} : \mathbf{y} = (x_1, x_2)\}. \tag{4}$$

And then define two further shift neighbourhoods without the origin as:

Input shape	ResNet101	Shift	Multi-shift	Flattened multi-shift
224×224	conv 7 × 7 stride-2, max-pool 3 × 3 stride-2, out: 64			
56×56	$\begin{bmatrix} \text{conv1} \times 1,\ 64 \\ \text{conv3} \times 3,\ 64 \\ \text{conv1} \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} \text{conv1} \times 1,\ 64 \\ \mathbf{S}\ \text{conv1} \times 1,\ 64 \\ \text{conv1} \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} \mathbf{S}\ \text{conv1} \times 1,\ 64 \\ \mathbf{S}\ \text{conv1} \times 1,\ 64 \\ \mathbf{S}\ \text{conv1} \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} \mathbf{S}\ \text{conv1} \times 1, 256 \\ \mathbf{S}\ \text{conv1} \times 1, 256 \\ \mathbf{S}\ \text{conv1} \times 1, 256 \end{bmatrix} \times 1$
56×56	$\begin{bmatrix} \text{conv1} \times 1, 128 \\ \text{conv3} \times 3, 128 \\ \text{conv1} \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} \text{conv1} \times 1, 128 \\ \mathbf{S}\ \text{conv1} \times 1, 128 \\ \text{conv1} \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} \mathbf{S}\ \text{conv1} \times 1, 128 \\ \mathbf{S}\ \text{conv1} \times 1, 128 \\ \mathbf{S}\ \text{conv1} \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} \mathbf{S}\ \text{conv1} \times 1, 512 \\ \mathbf{S}\ \text{conv1} \times 1, 512 \\ \mathbf{S}\ \text{conv1} \times 1, 512 \end{bmatrix} \times 1$
28×28	$\begin{bmatrix} \text{conv1} \times 1,\ 256 \\ \text{conv3} \times 3,\ 256 \\ \text{conv1} \times 1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} \text{conv1} \times 1,\ 256 \\ \mathbf{S}\ \text{conv1} \times 1,\ 256 \\ \text{conv1} \times 1,\ 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} \mathbf{S}\ \text{conv1} \times 1,\ 256 \\ \mathbf{S}\ \text{conv1} \times 1,\ 256 \\ \mathbf{S}\ \text{conv1} \times 1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} \mathbf{S}\ \text{conv1} \times 1, 1024 \\ \mathbf{S}\ \text{conv1} \times 1, 1024 \\ \mathbf{S}\ \text{conv1} \times 1, 1024 \end{bmatrix} \times 8$
14×14	$\begin{bmatrix} \text{conv1} \times 1,\ 512 \\ \text{conv3} \times 3,\ 512 \\ \text{conv1} \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} \text{conv1} \times 1,\ 512 \\ \mathbf{S}\ \text{conv1} \times 1,\ 512 \\ \text{conv1} \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} \mathbf{S}\ \text{conv1} \times 1,\ 512 \\ \mathbf{S}\ \text{conv1} \times 1,\ 512 \\ \mathbf{S}\ \text{conv1} \times 1,\ 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} \mathbf{S}\ \text{conv1} \times 1,\ 2048 \\ \mathbf{S}\ \text{conv1} \times 1,\ 2048 \\ \mathbf{S}\ \text{conv1} \times 1,\ 2048 \end{bmatrix} \times 1$
7×7	avg. pool 7 × 7, fc 1000, soft-max			

Table 1. Overview of the architectures used in this work. Repeating residual blocks are indicated in square brackets, with the number of times the block is repeated to the right. A bold **S** indicates shift placement. For each convolution in a residual block, the number after the comma indicates the number of output channels from an operation.

$$N_{8C-O}(\mathbf{x}) = N_{8C}(\mathbf{x}) \setminus N_O(\mathbf{x}), \tag{5}$$

$$N_{4C-O}(\mathbf{x}) = N_{4C}(\mathbf{x}) \setminus N_O(\mathbf{x}). \tag{6}$$

As a form of sanity check, we ask what happens if we use no shifts at all. This creates a baseline for the benefit of using shift operations when compared to a network of otherwise identical configuration. This case uses the 'no-shift' neighbourhood. Hence in total we investigate five neighbourhood variants. We apply these operations to the original ResNet [10]. We replace the 3×3 convolution within each residual block with a shift operation immediately followed by a point-wise 1×1 convolution. In all experiments (CIFAR100 and ImageNet) we use ResNet's 'bottleneck' residual block design (Table 1). The proposed changes are shown diagramatically in Fig. 1. We (initially) do not alter the channel structure of a block. We do this to ensure that each shift design is identical in terms of FLOPs and parameter count and can be more simply compared to ResNet.

Finally, we note a change to the downsampling method for shift-based networks. ResNet uses stride-2 spatial (3×3) convolutions to downsample within a residual block. All input activations to this spatial convolution then contribute to its output. For shift / point-wise convolution based residual blocks, this is no longer the case: most of an input activation map's information is lost following a (shifted) stride-2 point-wise convolution. We instead use a 2×2 average pooling to downsample, similar to [14]. We perform this pooling immediately prior to the shift / point-wise convolution, matching the downsample location to ResNet.

3.2. Multi-stage shifting residual blocks

In Fig. 1 b we simply replace the spatial convolution inside each residual block with a shift operations followed by 1×1 convolution. We now add further shift operations before the down-sampling and up-sampling 1×1 point-wise convolutions (Fig. 1 c), these convolutions previously intended to be used only for dimensionality reduction and expansion [10]. By adding shifts, we can add spatial information to these down- and up-sampling convolutions and expand the theoretical receptive field of each block significantly, as shown in Fig. 1 c. The same maximum receptive field extent of three blocks is then accomplished in one block, as information from a wider area is incorporated in each block's network optimization.

Inspired by [36] we also add an inner residual connection, which is across the middle convolution of the residual block. Now that spatial information is also carried by the first and last convolutions of the residual block, such an inner residual connection will no-longer carry only redundant information with respect to the outer residual connection.

3.3. Flattening the residual bottleneck

We now look at the network channel structure in this multiple shift setting. The purpose of down- and then up-sampling within bottlenecks is to reduce the dimensionality of the spatial convolution in each residual block [10]. While this process reduces the amount of information processed by the spatial convolution, it also reduces computational expense. By using shifts and 1×1 convolutions, we have removed spatial convolutions from network. Shift based networks then do not have the same computational need to perform dimensionality reduction. As such, we flatten ResNet's channel structure by widening the channel count in the middle of each block to be the same as the residual (see Table 1). This change is motivated by the improved performance of an increased channel width in other contexts, such as in Wide ResNet[35] and ResNeXT [33]. Even without spatial convolutions, this change increases the pa-

	CIFAR-100			ImageNet		
	#params	FLOPs	acc.	#params	FLOPs	acc.
ResNet101 [10]	1078K	154M	74.9	44.6M	7.80G	77.6
ResNet50 [10]	540K	78M	72.3	25.6M	4.09G	75.9
8-connected shift	605K	85M	74.3	25.6M	4.41G	77.3
4-connected shift	605K	85M	73.8	25.6M	4.41G	77.3
8-connected shift (nO)	605K	85M	74.2	25.6M	4.41G	77.0
4-connected shift (nO)	605K	85M	73.5	25.6M	4.41G	77.0
No shift	605K	85M	58.4	25.6M	4.41G	61.2

Table 2. Results for directly replacing spatial convolutions in ResNet101 with shifts of various neighbourhoods, compared to baselines of ResNet101 and ResNet50. nO denotes no origin. We find that 4-connected neighbourhoods are sufficient shifting in residual networks.

rameter count and FLOPs of the network. As the receptive field extent has also increased due to the multi-shift architecture, we reduce the length of the network to limit these costs to roughly the same as the original ResNet. Table 1 gives an overview of the architectures of this work.

4. Experiments and Discussion

Datasets: We focus on two well-known image recognition datasets: CIFAR-100 and ImageNet. CIFAR-100 contains 50,000 training examples and 10,000 test examples for 100 classes. All images are of size 32 × 32. For ImageNet, we use the 1,000 classes and 1.3M images train / 50K images test split as outlined by the Large-Scale Visual Recognition Challenge (ILSVRC) [25]. All images are resized to a resolution of 224 × 224.

Models and training: The initial ResNet model we take from [10]. For all datasets, we employ ResNet101 as a baseline, using the 'bottleneck' residual block. The same channel configuration as ResNet101 is used for all shift implementations; the computational cost is then identical across all shift based networks in the first experiments (Table 2).

To train ImageNet we use an initial learning rate of 0.1 and reduce it by a factor of 10 every 30 epochs for 100 epochs in total. We use a momentum of 0.9 and a batch size of 128. In training we use a random-resized crop and a single central crop for testing following [10]. We test one weight decay of 4×10^5 in the first set of ImageNet experiments (Table 2), and this value and an additional weight decay value of 1×10^4 in the second set of experiments (Table 3). Results for ImageNet are from training on 4 NVIDIA 1080Ti GPUs. Code and trained models are available online.

For training CIFAR-100, we use the same initial learning rate of 0.1 and reduce it by a factor of 10 every 100 epochs for 300 epochs. Training on a single NVIDIA TITANX GPU, we use a higher weight decay of 5×10^4, and a batch size also of 128.

4.1. Comparing shift operations

In table 2 we show how the direct replacement of spatial convolutions in ResNet with different shift types affects accuracy. We first look at CIFAR100 results. When compared to the ResNet101 baseline, all investigated shifts decrease computational cost by nearly half and suffer an accuracy penalty. This penalty is however smaller than that of using a shorter ResNet of comparable computational cost, such as ResNet50. The accuracy drop is also slightly larger for those shifts not including the origin than those shifts that do include the origin. This implies that, even though the residual connection carries information about the origin, it is still necessary to also include this information within shift operations. On CIFAR100, when using only one shift within a residual block, 8-connected shifts tend to outperform 4-connected shifts.

Similar results on are seen on ImageNet as on CIFAR-100: using shifts reduces computational cost, but an accuracy penalty is suffered. Noting that the absolute accuracies on both CIFAR-100 and ImageNet are similar, this penalty is smaller for ImageNet, between -0.3% and -0.6%, than for CIFAR-100, between -0.6% and -1.4%. We again find that shifts with an origin component outperform those without an origin component. One important difference between CIFAR-100 and ImageNet results is that 4C shifts show equal performance to 8C shifts. This result is unexpected, as the theoretical size of the receptive field is restricted for 4C shifts when compared to 8C shifts (Fig. 1).

Lastly, we note that for both CIFAR-100 and ImageNet, we find that having no shift at all drops accuracy significantly, but only to 58.4% and 61.2% for each dataset respectively. That network accuracy remains this high is surprising: these networks have only a single spatial convolution in their first layer. All other convolutions are point-wise 1 × 1 and cannot include spatial information (Table 1) - yet accuracy is still high enough to beat AlexNet [18]. Such no-shift networks are similar in structure to BagNets [2] - networks which have a highly restricted set of spatial convolutions. The most important distinction is that our no-origin networks do not include *any* spatial convolutions beyond the first convolution. Comparatively, BagNets still include one additional spatial convolution in each of ResNet's four layers. Our networks then have a greater spatial extent restriction than BagNets. Our results then suggest that perceptual tasks such as ImageNet can be solved by even smaller spatial feature extents than previous shown in [2].

In the next section, we examine the effects of placing shifts at additional positions in the network. We do this for both the best performing shifts on ImageNet from this section, the 8C and 4C shifts including an origin component.

	CIFAR-100			**ImageNet**			
	#params	FLOPs	accuracy	#params	FLOPs	accuracy	
						wd: 4×10^{-5}	wd: 1×10^{-4}
Baselines							
ResNet101 [10]	1078K	154M	74.9	44.6M	7.80G	77.6	77.4
Multi-shifting							
8-connected	605K	85M	74.3	25.6M	4.41G	76.8	77.2
4-connected	605K	85M	75.1	25.6M	4.41G	77.3	77.6
Flattened architecture							
8-connected	1068K	162M	76.9	40.8M	7.72G	77.2	77.8
4-connected	1068K	162M	77.5	40.8M	7.72G	77.8	78.4

Table 3. Results for networks with additional shifts placed before down- and up-sampling convolutions, compared to the baseline ResNet101. For both datasets we find that, when using multiple shifts, 4-connected shifts are preferred over 8-connected shifts. The accuracies of multiple 4-connected shift networks are competitive with the baseline at a reduced computational cost. Using multiple shifts in a flattened residual block channel structure results in an improved performance over standard ResNets at a similar computational cost. In this flattened architecture, we again find 4-connected shifts are preferred over 8-connected shifts.

4.2. Multi-stage shifting

In Table 3 we show networks with multiple shift operations in each residual block and compare them to a baseline ResNet101, again on both CIFAR-100 and ImageNet.

For CIFAR-100, we find that the multi-4C shift networks improves against single-4C shift networks (+1.2%), but multi-8C shift networks show no improvement over single-8C shift networks. The accuracy for multi-4C shift networks is slightly above the baseline (+0.2%), while reducing computational costs by 45%. The final architecture studied flattens the channel structure of bottlenecks and has a reduced network length, keeping computational costs approximately the same as the baseline ResNet101. In this architecture, 4C shifts are again found to outperform 8C shifts. Both shift types outperform the baseline in this architecture, with 4C shifts giving the greatest accuracy improvement (+2.6%).

For ImageNet models we compare two weight decay settings: 4×10^{-5}, suggested in [13] for use with ResNet architectures, and 1×10^{-4}, used in the original ResNet experiments [10]. We find that multi-shift networks are particularly sensitive to weight decay within this range. All multi-shift networks benefit from using the same weight decay as originally suggested for ResNet, though ResNet itself does not. While not shown in Table 2, a higher weight decay degrades performance for single shifts. In both weight decay settings, we find that multiple 4C shifts outperform multiple 8C shifts. This is despite the reduced theoretical receptive field size of 4C shifts when compared to 8C shifts. Comparing optimal weight decay settings for each network, adding multiple shift modules improves 4C shift results (+0.3%), but does not change 8C results. The multi-4C shift architecture provides the same accuracy as the original ResNet101, yet with a 43% reduction in computational costs.

Table 3 also shows ImageNet results from networks with a flattened channel structure and equipped with either multiple 4C shifts or multiple 8C shifts. We also find that in this architecture, multiple 4C shift out-perform multiple 8C shifts. In this architecture, using either 8C or 4C shifts results in an improved accuracy against the baseline ResNet101 while keeping computational cost approximately the same, with use of 4C shifts yielding the largest improvement (+0.8%). This improved accuracy is in spite of these shift-based networks being considerably less deep (35 layers) than the baseline (101 layers), see Table 1. This choice of depth was made to keep the FLOPs and parameter count approximately the same as ResNet, and does not appear to have restricted accuracy.

In our final figure, Fig. 2, we comparatively evaluate the top1-accuracy on ImageNet of different networks as a function of both the number of FLOPs and the number of network parameters. The figure shows how our multi-4C shift residual network design significantly improves in computational cost against one of the most popular modern network designs, ResNet [10], while maintaining accuracy. On the other hand, our flattened multi-4C shift architecture has a similar numbers of FLOPs and parameters as ResNet and improves accuracy.

We draw a comparison to shift papers with ImageNet architectures [16, 32, 31]. We highlight that while the networks shown in these works are computationally efficient, their accuracies are comparatively lower. This is as these works principally focused on improving compact, low parameter / FLOP networks; their results can thus be seen towards the left of the figure. The exception is ShiftResNet50 shown in [32]. The exact architecture and FLOPs of this network were not reported; here we have estimated the network's FLOPs from the number of reported parameters and show the results in Fig. 2.

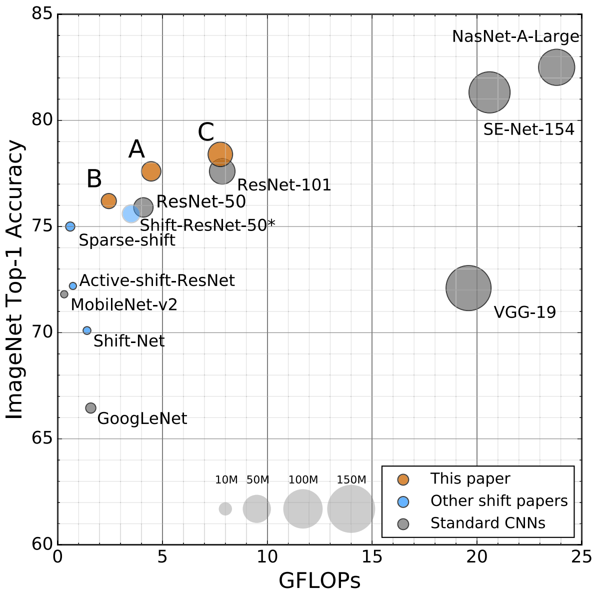

Figure 2. ImageNet top-1 accuracies as they relate to FLOPs, with the parameter count indicated by circle size. See text for data sources. Our approach (orange circles) demonstrates shifts can improve FLOPs/parameters or accuracy against ResNet. 'A' and 'B' denote 4C-MS-ResNet101 and 4C-MS-ResNet50. Both are models using multiple shifts with the original ResNet channel structure. 'C' denotes 4C-MSF-ResNet35 which uses the flattened channel structure. All variants use 4-connected shifts. In terms of accuracy, our networks outperform the popular CNN architectures VGG-19 [28] and MobileNetv2 [26] and all other shift-based networks [16, 32]. The FLOPs for Shift-ResNet-50 [32] are not available and have been estimated from the parameter count.

We also compare to well established standard CNN architectures [29, 28, 13, 37, 26]. For the accuracies, FLOPs and parameters of standard CNNs, we use the benchmark analysis of Bianco et al. [1]. Compared to other standard CNN architecures, the accuracy of our networks are superior to MobileNetv2 [26], GoogleNet [29] and VGG [28]. The current best performers on ImageNet, SENet-154 [13] and NasNet-A-Large [37], have a higher accuracy than our networks, but come with a much larger FLOP and parameter demand. We envision that these networks can similarly benefit from using 4-connected shifts in their architecture, reducing their FLOP requirement yet maintaining accuracy.

5. Conclusions

This work investigates shifts in deep residual networks and how best to apply them in the high accuracy, large image classification setting. We examine shifts based on both the 8-connected and 4-connected neighbourhoods. We find that, when used solely within residual blocks, both neighbourhoods offer similar performance. When used multiple times, the shift neighbourhood should be restricted to the 4-connected neighbours. As such, we posit that only shifting to the 4 nearest neighbours is sufficient in deep residual networks. We have outlined two high-accuracy networks using 4-connected shifts: the first reduces computational cost against ResNet101 by 43% without compromising on accuracy; the second improves on ResNet101's accuracy, while keeping computational costs roughly equal. These results show that shifts can be successfully applied in the high-accuracy deep learning setting, offering large improvements in computational cost or accuracy. Code and trained models are available online.

References

[1] Simone Bianco, Remi Cadene, Luigi Celona, and Paolo Napoletano. Benchmark analysis of representative deep neural network architectures. *IEEE Access*, 6:64270–64277, 2018. 7

[2] Wieland Brendel and Matthias Bethge. Approximating cnns with bag-of-local-features models works surprisingly well on imagenet. In *ICLR*, 2019. 5

[3] Nadav Cohen, Or Sharir, and Amnon Shashua. On the expressive power of deep learning: A tensor analysis. In *COLT*, 2016. 2

[4] Emily L. Denton, Wojciech Zaremba, Joan Bruna, Yann LeCun, and Rob Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. In *NeurIPS*. 2014. 2

[5] Luigi Di Stefano and Andrea Bulgarelli. A simple and efficient connected components labeling algorithm. In *ICIAP*, pages 322–327, 1999. 3

[6] M.J.B. Duff, D.M. Watson, T.J. Fountain, and G.K. Shaw. A cellular logic array for image processing. *Pattern Recognition*, 5(3):229 – 247, 1973. 1

[7] Martín Abadi et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems, 2016, arXiv:1603.04467. 3

[8] Yiwen Guo, Anbang Yao, and Yurong Chen. Dynamic network surgery for efficient dnns. In *NeurIPS*, 2016. 2

[9] Song Han, Jeff Pool, John Tran, and William J. Dally. Learning both weights and connections for efficient neural networks. In *NeurIPS*, 2015. 2

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1, 4, 5, 6

[11] Yihui He, Xianggen Liu, Huasong Zhong, and Yuchun Ma. Addressnet: Shift-based primitives for efficient convolutional neural networks. In *WACV*, 2019. 1, 2, 3

[12] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *NeurIPS workshop*, 2015. 2

[13] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, 2018. 1, 6, 7

[14] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 4

[15] Forrest N. Iandola, Matthew W. Moskewicz, Khalid Ashraf, Song Han, William J. Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and less than 1mb model size, 2016, arXiv:1602.07360. 2

[16] Yunho Jeon and Junmo Kim. Constructing fast network through deconstruction of convolution. In *NeurIPS*, 2018. 1, 2, 6, 7

[17] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009. 1, 2

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012. 1, 5

[19] Hsiang-Tsung Kung, Bradley McDanel, and Sai Quian Zhang. Mapping systolic arrays onto 3d circuit structures: Accelerating convolutional neural network inference. In *SiPS*, 2018. 2

[20] Myunggi Lee, Seungeui Lee, Sungjoon Son, Gyutae Park, and Nojun Kwak. Motion feature network: Fixed motion filter for action recognition. In *ECCV*, 2018. 2

[21] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NeurIPS workshop*, 2017. 3

[22] Gerhard Ritter and Joseph Wilson. *Handbook of Computer Vision Algorithms in Image Algebra*. CRC Press, Inc., 1996. 1, 3

[23] Jos Roerdink and A Meijster. The watershed transform: Definitions, algorithms and parallelization strategies. *Fundamental Informatica*, 41, 10 2003. 3

[24] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Y Bengio. Fitnets: Hints for thin deep nets, 2014,arXiv:1412.6550. 2

[25] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015. 1, 2, 5

[26] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018. 2, 7

[27] Jean Serra. *Image Analysis and Mathematical Morphology*. Academic Press, Inc., 1983. 1

[28] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 7

[29] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 7

[30] Lucas J. van Vliet and Ben J.H. Verwer. A contour processing method for fast binary neighbourhood operations. *Pattern Recognition Letters*, 7(1):27 – 36, 1988. 3

[31] Chen Weijie, Di Xie, Yuan Zhang, and Shiliang Pu. All you need is a few shifts: Designing efficient convolutional neural networks for image classification. In *CVPR*, 2019. 2, 6

[32] Bichen Wu, Alvin Wan, Xiangyu Yue, Peter Jin, Sicheng Zhao, Noah Golmant, Amir Gholaminejad, Joseph Gonzalez, and Kurt Keutzer. Shift: A zero flop, zero parameter alternative to spatial convolutions. In *CVPR*, 2018. 1, 2, 3, 6, 7

[33] Saining Xie, Ross Girshick, Piotr Dollar, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017. 4

[34] Yifan et al. Yang. Synetgy: Algorithm-hardware co-design for convnet accelerators on embedded fpgas. In *International Symposium on Field-Programmable Gate Arrays*, 2018. 2

[35] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016. 4

[36] Ke Zhang, Miao Sun, Tony X. Han, Xingfang Yuan, Liru Guo, and Tao Liu. Residual networks of residual networks: Multilevel residual networks. *TCSVT*, 28(6), 2018. 4

[37] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. Learning transferable architectures for scalable image recognition. In *CVPR*, 2018. 1, 7