Supplementary material: Truly shift-invariant convolutional neural networks

A. Non-linear activation functions and shift invariance

We saw in Section 3.2 of the paper that anti-aliasing a signal before downsampling restores sum-shift-invariance. In particular, consider a 1-D signal $x_0(n)$ and its 1-pixel shift $x_1(n) = x_0(n-1)$. Anti-aliasing the two signals (with an ideal low pass filter) followed by downsampling with stride 2 results in $y_0^a(n)$ and $y_1^a(n)$ with DTFTs

$$Y_0^a(\omega) = \frac{X_0(\omega/2)}{2},\ Y_1^a(\omega) = \frac{X_0(\omega/2)e^{-j\omega/2}}{2}, \quad (1)$$

that satisfy $Y_0^a(0) = Y_1^a(0)$. Azulay and Weiss pointed out in [1] that the sum-shift invariance obtained via anti-aliasing is lost due to the action of non-linear activation functions like ReLU in convolutional neural networks. They postulated that this happens through the generation of high-frequency content after applying ReLU. We elaborate on this phenomenon here and also show that high frequencies alone do not provide a full picture.

Let $g(\cdot)$ be a generic pointwise non-linear activation function applied to the outputs of anti-aliased downsampling. Owing to the pointwise nature of g, the stride operation and the non-linearity can be interchanged, making the network block in Fig. 1(a) equivalent to the one in Fig. 1(b). Notice in Fig. 1(b) that despite anti-aliasing x_0 with an ideal low pass filter LPF, g generates additional high frequencies which can result in aliasing on downsampling. One can not simply use another low pass filter to get rid of these newly generated aliased components. For example, a new low pass filter block added after g in Fig. 2(a) can be interchanged with the stride operation to result in a dilated filter which is not low pass any more (Fig. 2(b)).

While high frequencies generated by non-linear activations can lead to invariance loss for various choices of g, we show in Section A.1 that this might not always be necessary. For example, polynomial activations, despite generating aliased components, do not impact sum-shift-invariance. Therefore, in addition to its high frequency generation ability, we also take a closer look at how the ReLU non-linearity affects sum-shift invariance in terms of its thresholding behavior in Section A.2.

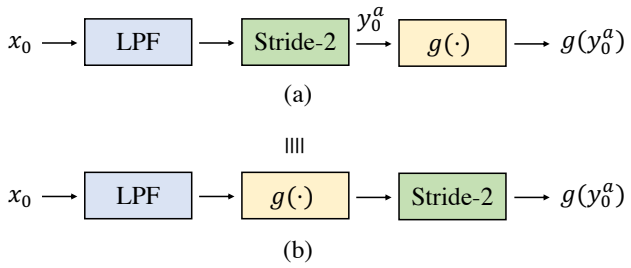

Figure 1. Pointwise non-linearity g can be interchanged with the stride operation. Despite anti-aliasing x_0 with LPF block, g generates high frequencies which can lead to additional aliasing during downsampling.

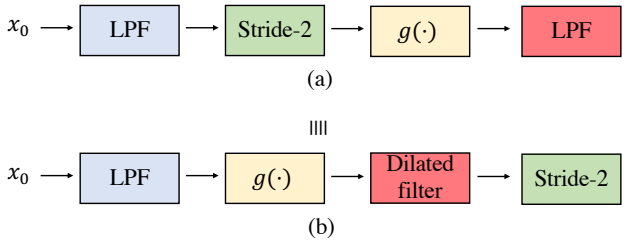

Figure 2. Additional low pass filtering after g in (a) does not eliminate the impact of aliasing. This is because, as shown in (b), interchanging the final LPF block with stride operation, results in a dilated version of the filter which is not low-pass any more.

A.1. Action of polynomial non-linearities on sum-shift invariance

In Theorem 1 from Section 3.2 in the paper, we stated that for any integer $m > 1$, non-linear activation functions of the form $g(y) = y^m$ do not impact sum-shift-invariance. We provide the proof below.

Proof. Let the DTFTs of $z_0 = g(y_0^a)$ and $z_1 = g(y_1^a)$ be $Z_0(\omega)$ and $Z_1(\omega)$. Then by definition of the DTFT,

$$Z_0(0) = \sum_{n\in\mathbb{Z}} z_0(n), \text{ and } Z_1(0) = \sum_{n\in\mathbb{Z}} z_1(n). \quad (2)$$

Since $z_0 = (y_0^a)^m$, and $z_1 = (y_1^a)^m$, we have

$$Z_0(\omega) = \Big(\underbrace{Y_0^a(\omega) \circledast Y_0^a(\omega) \circledast \ldots \circledast Y_0^a(\omega)}_{\text{m times}} \Big), \quad (3)$$

$$Z_1(\omega) = \Big(\underbrace{Y_1^a(\omega) \circledast Y_1^a(\omega) \circledast \ldots \circledast Y_1^a(\omega)}_{\text{m times}} \Big), \quad (4)$$

where $\circledast$ represents circular convolution. For $i \in \{0, 1\}$, we can write

$$Z_i(\omega) = \Big(\frac{1}{2\pi}\Big)^{m-1} \int_{-\pi}^{\pi} \cdots \int_{-\pi}^{\pi} Y_i^a(\alpha_1) \cdots Y_i^a(\omega - \sum_{i=1}^{m-1} \alpha_i) d\bar{\alpha}, \quad (5)$$

where $\bar{\alpha} = (\alpha_1, \alpha_2, \ldots \alpha_{m-1})$. From (1), we have

$$Y_1^a(\omega) = Y_0^a(\omega) e^{-j\omega/2}. \quad (6)$$

Using (5) and (6), we can write $Z_1(\omega) = Z_0(\omega) e^{-\frac{j\omega}{2}}$, which when combined with (2) gives

$$\sum_{n \in \mathbb{Z}} z_0(n) = \sum_{n \in \mathbb{Z}} z_1(n). \quad (7)$$

□

Using linearity of Fourier transform, the result in Theorem 1 can be extended to arbitrary polynomial activation functions of the form $g(y) = \sum_{i=0}^{m} a_i y^i$ with $m > 1$.

A.2. ReLU spoils sum-shift-invariance

We now consider the ReLU non-linear activation function, $h(y) = \text{relu}(y)$, which clips all negative values of signal y to zero. Unlike the case with polynomials in Section A.1, deriving a closed form expression for the DTFTs of $h(y_0^a)$ and $h(y_1^a)$, for arbitrary x_0 and x_1 is non-trivial. We therefore analyze a simpler case where x_0 is assumed to be a cosine signal, and illustrate how sum-shift invariance is lost due to ReLUs.

Let x_0 be an N length 1-D cosine and $x_1 = x_0(n-1)$ be its 1-pixel shift. We define the two signals as

$$x_0 = \cos\Big(\frac{2\pi n}{N}\Big), \text{ and } x_1 = \cos\Big(\frac{2\pi(n-1)}{N}\Big) \quad (8)$$
$$n \in \{0, 1, \ldots N-1\}.$$

For any $N > 4$, x_0 satisfies the Nyquist criterion and is anti-aliased by default. For $N' = N/2$ and $n \in \{0, 1, \ldots N'-1\}$, the downsampled outputs y_0^a and y_1^a are then given by

$$y_0^a(n) = x_0(2n) = \cos\Big(\frac{2\pi n}{N'}\Big), \quad (9)$$

$$y_1^a(n) = x_1(2n) = \cos\Big(\frac{2\pi(n-1/2)}{N'}\Big). \quad (10)$$

Note that y_0^a and y_1^a are structurally similar signals, and can be interpreted as *half-pixel* shifted versions of each other. The action of h on y_i^a can be regarded as multiplication by a window which is zero for any n where $y_i^a(n) < 0$. We construct sets $\{S_i^+\}_{i=0}^1$ containing n where $y_i^a(n) > 0$. For simplicity in constructing the sets, we assume $N' > 6$ and divisible by 4 (similar conclusions from below can be reached without these simplifying assumptions as well). Then we have

$$S_0^+ = \Big\{ n : n \in \mathbb{Z}, n \in \Big[0, \frac{N'}{4} - 1\Big] \cup \Big[\frac{3N'}{4} + 1, N' - 1\Big] \Big\}, \quad (11)$$

$$S_1^+ = \Big\{ n : n \in \mathbb{Z}, n \in \Big[0, \frac{N'}{4}\Big] \cup \Big[\frac{3N'}{4} + 1, N' - 1\Big] \Big\}. \quad (12)$$

Notice that the supports S_0^+ and S_1^+ differ by 1 pixel near $n = \frac{N'}{4}$. This is because despite being structurally similar, y_0^a and y_1^a have slightly different zero crossings, which results in some differences in the support of thresholded outputs. We can now compute the sums $\sum h(y_0^a)$ and $\sum h(y_1^a)$.

$$\sum_{n \in \mathbb{Z}} h(y_0^a)(n) = \sum_{n \in S_0^+} \cos\Big(\frac{2\pi n}{N'}\Big) \quad (13)$$

$$= \text{Re}\Big(\sum_{n \in S_0^+} e^{j\frac{2\pi n}{N'}} \Big) \quad (14)$$

$$= \frac{\cos(2\pi/N)}{\sin(2\pi/N)}. \quad (15)$$

Similarly, $\sum_{n \in \mathbb{Z}} h(y_1^a)(n)$ is given by

$$\sum_{n \in \mathbb{Z}} h(y_1^a)(n) = \text{Re}\Big(\sum_{n \in S_1^+} e^{\frac{j2\pi(n-1/2)}{N'}} \Big) \quad (16)$$

$$= \text{Re}\Big(e^{-\frac{j\pi}{N'}} \sum_{n \in S_1^+} e^{j\frac{2\pi n}{N'}} \Big). \quad (17)$$

We can rewrite (17) in terms of (14), and get

$$\sum_{n \in \mathbb{Z}} h(y_1^a)(n) \quad (18)$$

$$= \text{Re}\Big(e^{-\frac{j\pi}{N'}} \sum_{n \in S_0^+} e^{j\frac{2\pi n}{N'}} + e^{-\frac{j\pi}{N'}} e^{\frac{j2\pi n}{N'}} |_{n=N'/4} \Big) \quad (19)$$

$$= \cos\Big(\frac{2\pi}{N}\Big) \sum_{n \in \mathbb{Z}} h(y_0^a)(n) + \sin\Big(\frac{2\pi}{N}\Big). \quad (20)$$

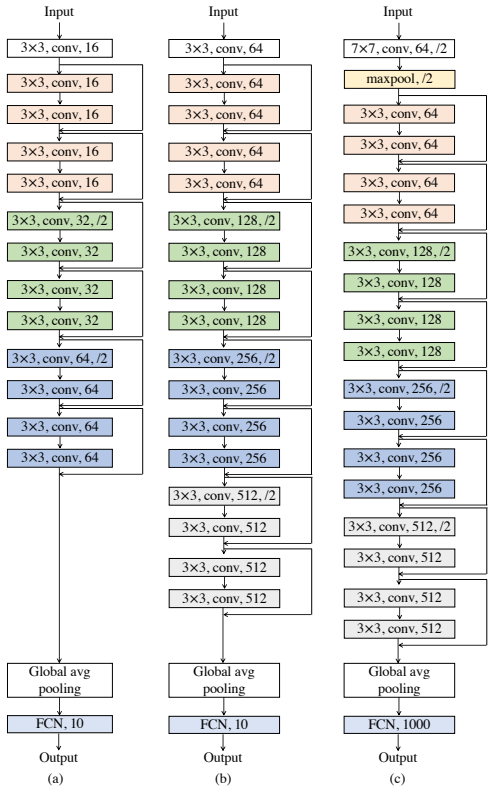

Figure 3. Illustration of baseline ResNet architectures used in our experiments. (a) ResNet-20, (b) ResNet-18 used in CIFAR-10 classification. (c) Baseline ResNet-18 used in the ImageNet classification experiments.

(20) illustrates the loss in sum-shift-invariance caused by ReLU. Notice that the differences in $\sum h(y_0^a)$ and $\sum h(y_1^a)$ arise due to minor differences in the signal content in y_0^a and y_1^a, which are amplified by ReLU. The term $sin(2\pi/N)$ arises due to a 1-pixel difference in the supports of $h(y_0^a)$ and $h(y_1^a)$, whereas the cosine term is associated with $e^{-j\omega/2}$ from (1), again depicting the impact of small differences in y_0^a and y_1^a.

B. Implementation details

We trained ResNet models with APS, anti-aliasing and baseline conventional downsampling approaches on CIFAR-10 and ImageNet datasets, and compared their achieved classification consistency and accuracy. For CIFAR-10 experiments, four variants of the architecture were used: ResNet-20, 56, 18 and 50. ResNet 20 and 56 were originally introduced in [2] for CIFAR-10 classification and are smaller models with number of channels: $\{16, 32, 64\}$ in different layers, and use stride 2 twice, which results in a resolution of 8×8 in the final convolutional feature maps. On the other hand, ResNet-18 and 50 contain $\{64, 128, 256, 512\}$ number of channels, and downsample three times with a stride 2, resulting in final feature map resolution of 4×4. Similar to the experiments with CIFAR-10 in [2], we use a convolution with stride 1 and kernel size of 3×3 in the first convolutional layer. In all architectures, global average pooling layers are used at the end of the convolutional part of the networks. Fig. 3(a)-(b) illustrate the baseline architectures of ResNet-20 and 18 used in our experiments.

The original training set of the CIFAR-10 dataset was split into training and validation subsets of size 45k and 5k. All models were trained with batch size of 256 for 250 epochs using stochastic gradient descent (SGD) with momentum 0.9 and weight decay $5\mathrm{e}{-4}$. The initial learning rate was chosen to be 0.1 and was decayed by a factor of 0.1 every 100 epochs. Training was performed on a single NVIDIA V-100 GPU. All the models were randomly initialized with a fixed seed before training. The models with the highest validation accuracy were used for evaluation on the test set.

For ImageNet classification, we used standard ResNet-18 model as baseline whose architecture is illustrated in Fig. 3(c). In all experiments, input image size of 224×224 was used. The models were trained with batch size of 256 for 90 epochs using SGD with momentum 0.9 and weight decay of $1\mathrm{e}{-4}$. An initial learning rate of 0.1 was chosen which was decayed by a factor of 0.1 every 30 epochs. The models were trained in parallel on four NVIDIA V-100 GPUs. We report results for models with the highest validation accuracy.

We were able to show significant improvements in consistency and accuracy with APS over baseline and anti-aliased downsampling without substantial hyper-parameter tuning. Further improvements in the results with better hyper-parameter search are therefore possible.

B.1. Embedding APS in ResNet architecture

We replace the baseline stride layers in the ResNet architectures with APS modules. To ensure shift invariance, a consistent choice of polyphase components in the main and residual branch stride layer is needed. APS uses a permutation invariant criterion (like argmax) to choose the component to be sampled in the main branch. The index of the chosen component is passed to the residual branch where the polyphase component with the same index is sampled. An illustration is provided in Fig. 4.

C. Impact of polyphase component selection method on classification accuracy

In the paper, we saw that APS achieves perfect shift invariance by selecting the polyphase component with the

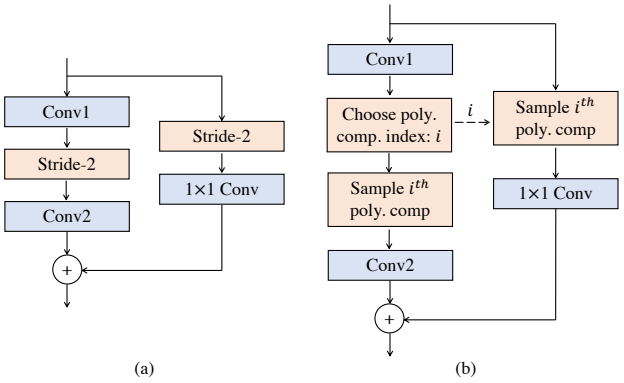

Figure 4. Residual connection block with (a) baseline stride, (b) APS layer.

highest l_2 norm, i.e.

$$y_{\text{APS}} = y_{i_1 j_1}, \quad \text{where } i_1, j_1 = \underset{i,j}{\text{argmax}}\{\|y_{ij}\|_2\}_{i,j=0}^{1}. \tag{21}$$

This can also be achieved, however, with other choices of shift invariant criteria. Here, we study the impact of different such criteria on the accuracy obtained on CIFAR-10 classification. In particular, we explore maximization of l_p norms with $p = 1$ and ∞ in addition to $p = 2$. We also consider minimization of l_1 and l_2 norms. We run the experiments on ResNet-18 architecture with 9 different initial random seeds and report the mean and standard deviation of achieved accuracy on the test set.

Table 1 shows that choosing the polyphase component with the largest l_∞ norm provides the highest classification accuracy which is then followed by choosing the one with the highest l_2 norm and l_1 norm. Additionally, the accuracy obtained when choosing polyphase component with minimum l_2 norm is somewhat lower than the case which chooses maximum l_2 norm. We believe this could be due to the polyphase components with higher energy containing more discriminative features.

Note that for all cases in Table 1, the achieved classification accuracy is $\sim 2\%$ higher than that of baseline ResNet-18 (reported in the paper). This is because in each case, APS enables stronger generalization via perfect shift invariance prior.

D. Experiments with data augmentation

We saw in Section 4.1 of the paper that APS results in 100% classification consistency and more than 2% improvement in accuracy on CIFAR-10 dataset for models trained without any random shifts (data augmentation).

APS criterion	Accuracy	Consistency
argmax (l_1)	$93.89 \pm 0.27\%$	**100%**
argmax (l_2)	$94.03 \pm 0.26\%$	**100%**
argmax (l_∞)	$\mathbf{94.14 \pm 0.25\%}$	**100%**
argmin (l_1)	$93.92 \pm 0.12\%$	**100%**
argmin (l_2)	$93.90 \pm 0.16\%$	**100%**

Table 1. Impact of polyphase component selection method used by APS on CIFAR-10 classification accuracy.

Here, we assess how baseline sampling compares with APS when the models are trained on CIFAR-10 dataset with data augmentation (labelled as DA). The results are reported in Table 2.

We observe that while data augmentation does improve classification consistency for baseline models, it is still lower than APS which yields perfect shift invariance. Classification accuracy, on the other hand, for both the baseline and APS is comparable (within the limits of training noise) when the models are trained with random shifts. This is not surprising because data augmentation is known to improve classification accuracy on images with patterns similar to the ones seen in training set. Note that, as reported in Section 4.2 of the paper, accuracy of networks with APS is more robust to image corruptions, and the models continue to yield 100% classification consistency on all image distributions.

E. Downsampling circularly shifted images with odd dimensions

With circular shift, pixels that exit from one end of a signal roll back in from the other, thereby preventing any information loss. While this makes circular shifts convenient for evaluating the impact of downsampling on shift invariance over finite length signals, they can lead to additional artifacts at the boundaries when sampling odd-sized signals. For example, as illustrated in Fig. 5, while the polyphase components y_1 and $\tilde{y}_0$ are identical, y_0 and $\tilde{y}_1$ do not contain the same pixels near the boundaries. This is because downsampling an odd-sized signal with stride-2 breaks the periodicity associated with circular shifts, resulting in minor differences in the sets of polyphase components near the boundaries.

We investigate the impact of these artifacts by training ResNet-18 models with different downsampling modules on CIFAR-10 dataset with images center-cropped to size 30×30. These images result in odd-sized feature maps inside the networks which generate boundary artifacts after downsampling. The models were then evaluated on 30×30 center-cropped CIFAR-10 test set. Results in Table 3 show

Accuracy (unshifted)			Consistency	
Model	ResNet-18	ResNet-50	ResNet-18	ResNet-50
Baseline	91.96%	90.05%	90.88%	88.96%
APS-3	94.53%	93.80%	**100%**	**100%**
Baseline + DA	94.33%	**94.77%**	97.84%	97.64%
APS-3 + DA	**94.61%**	94.39%	**100%**	**100%**

Table 2. Impact of APS on classification consistency and accuracy (evaluated on unshifted images) obtained using models trained with random shifts in data augmentation. Models trained without data augmentation are also shown for reference.

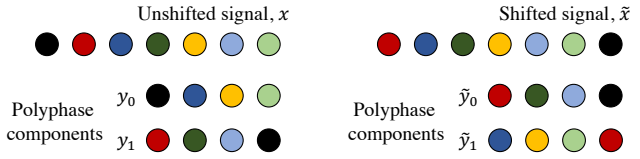

Figure 5. Boundary artifacts associated with circular shifts. Downsampling an odd length signal and its circular shift can result in minor differences in polyphase components near their boundaries.

that despite the presence of artifacts, both the classification consistency and accuracy on unshifted images is greater for models that use APS.

F. Timing analysis

APS computes the norms of polyphase components for downsampling consistently to shifts. This leads to a modest increase in the time required to perform a forward pass in comparison with baseline network. For example, a forward pass on a 224×224 image with a circular padded baseline ResNet-18 takes 8.15 ± 0.47ms on a single V-100 GPU. In comparison, ResNet-18 with APS layers takes 11.88 ± 0.06ms.

References

[1] Aharon Azulay and Yair Weiss. Why do deep convolutional networks generalize so poorly to small image transformations? *Journal of Machine Learning Research*, 20(184):1–25, 2019.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

	Baseline	APS	LPF-2	APS-2	LPF-3	APS-3	LPF-5	APS-5
Consistency	88.23%	98.13%	94.28%	98.48%	96.15%	98.65%	98.02%	**99.26%**
Accuracy (unshifted images)	90.91%	93.99%	93.22%	93.83%	93.56%	**94.34%**	94.28%	94.22%

Table 3. Classification consistency and accuracy obtained with different variants of ResNet-18 when evaluated on CIFAR-10 test set with images cropped to size 30×30. The models were trained without seeing random shifts during training. Despite the presence of boundary effects caused by circular shifts on odd-sized feature maps, we observe higher consistency and accuracy with models containing APS.