Truly shift-invariant convolutional neural networks

Anadi Chaman
University of Illinois at Urbana-Champaign
achaman2@illinois.edu

Ivan Dokmanić
University of Basel
ivan.dokmanic@unibas.ch

Abstract

Thanks to the use of convolution and pooling layers, convolutional neural networks were for a long time thought to be shift-invariant. However, recent works have shown that the output of a CNN can change significantly with small shifts in input—a problem caused by the presence of downsampling (stride) layers. The existing solutions rely either on data augmentation or on anti-aliasing, both of which have limitations and neither of which enables perfect shift invariance. Additionally, the gains obtained from these methods do not extend to image patterns not seen during training. To address these challenges, we propose adaptive polyphase sampling (APS), a simple sub-sampling scheme that allows convolutional neural networks to achieve 100% consistency in classification performance under shifts, without any loss in accuracy. With APS, the networks exhibit perfect consistency to shifts even before training, making it the first approach that makes convolutional neural networks truly shift-invariant.

1. Introduction

The output of an image classifier should be invariant to small shifts in the image. For a long time, convolutional neural networks (CNNs) were simply assumed to exhibit this desirable property [36, 37, 38, 39]. This was thanks to the use of convolutional layers which are shift equivariant, and non-linearities and pooling layers which progressively build stability to deformations [6, 42]. However, recent works have shown that CNNs are in fact not shift-invariant [2, 58, 18, 34, 31]. Azulay and Weiss [2] show that the output of a CNN trained for classification can change with a probability of 30% with merely a one-pixel shift in input images. Related works [31, 34] have also revealed that CNNs can encode absolute spatial location in images: a consequence of a lack of shift invariance.

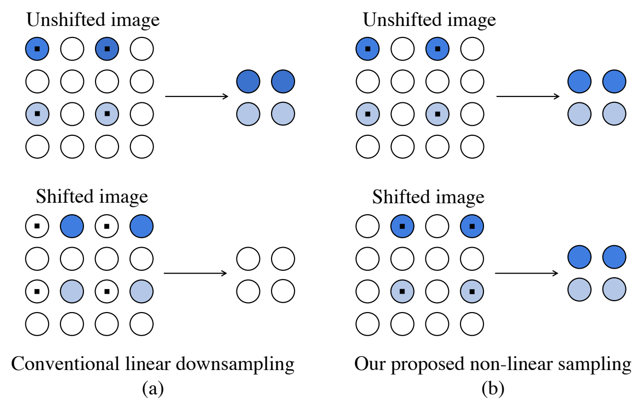

Figure 1. (a) Conventional downsampling is not robust to shifts. It samples image pixels at fixed locations on the grid (shown with small squares). Shifting the image changes pixel intensities located on the fixed grid, resulting in a different subsampled output. (b) By choosing the sampling grid that supports pixels with highest energy, our approach results in shift invariance.

One of the key reasons why CNNs are not shift-invariant is downsampling[1] [2, 58], or stride, which is a linear operation that samples evenly spaced image pixels located at fixed positions on the grid and discards the rest. As shown in Fig. 1(a), the results of downsampling an image and its shifted version can be significantly different. This is because shifting an image can change the pixel intensities located over the sampling grid. Various measures have been proposed in literature to counter this problem. With data augmentation [48], the output of a CNN can be made more robust to shifts by training it on randomly shifted versions of input images [2, 58]. This, however, improves the network's invariance only for image patterns seen during training [2]. Anti-aliasing or blurring spreads sharp image features across their neighbouring pixels which improves structural similarity between subsampled outputs of an image and its shifted version (Fig. 2(a)-(c)). One instance of this technique are strided average pooling layers [38]. Azulay and Weiss [2] showed that anti-aliasing

Code available at https://github.com/achaman2/truly_shift_invariant_cnns.

[1] Layers like strided max-pooling in CNNs can be regarded as a combination of a dense max-pooling operation followed by downsampling.

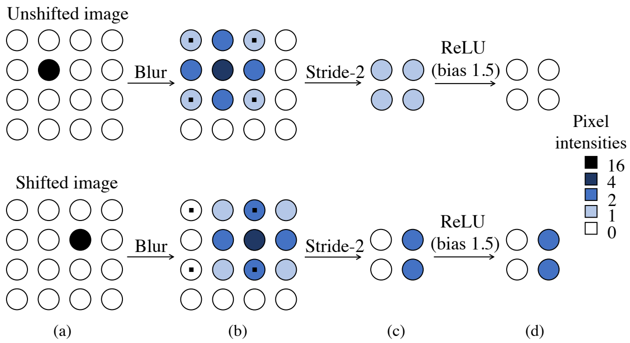

Figure 2. Anti-aliasing based solutions. (a) An image and its shift. (b)-(c) Blurring spreads sharp features across their neighbouring pixels. For the 2 images, the resulting pixel intensities over the sampling grid become more similar improving shift invariance. (d) Due to minor differences in the subsampled outputs in (c), they are thresholded differently by ReLU and shift invariance is lost.

a linear convolutional network with global pooling in the end completely restores shift invariance. Zhang [58] combined a dense max pooling layer and strided blurred pooling to boost shift invariance and accuracy of classification. While blurring-based methods do improve the network's robustness to shifts, they only achieve partial shift invariance. Their performance is limited by the action of non-linear activation functions such as ReLU [2]; we illustrate it in Fig. 2(d). Additionally, anti-aliasing beyond a certain point can result in over-blurring and loss of accuracy [58].

Taking inspiration from the recent works of Azulay and Weiss [2] and Zhang [58], we propose adaptive polyphase sampling (APS)—a simple non-linear downsampling scheme that provides substantial improvement in classification consistency over blurring-based methods. In fact, with only mild changes in the padding used by convolutional layers we show that APS enables 100% consistency in classification performance under shifts without any loss in accuracy, making it the first approach that makes CNNs *truly shift-invariant*.

APS achieves the aforementioned performance by addressing the root cause of the problem—the use of a fixed grid by current architectures to subsample an image, when there actually exist multiple sampling grids to choose from (Fig. 1(b)). The key idea used by APS is that shifting an image simply translates its pixels from one grid to the other. Therefore, instead of always choosing a fixed sampling grid, one can select the grid adaptively—for example by choosing the one which supports pixels with the highest energy. The resulting subsampled outputs are then identical for any shift in input.

With APS, the achieved boost in shift invariance is completely unaffected by the non-linear ReLU activations (or any other point-wise activations) in the network, and the improved robustness to shifts is consistent regardless of whether the network is tested on in- or out-of-distribution samples. Furthermore, CNNs using APS exhibit complete shift invariance even before training, implying that the invariance prior is truly embedded in the architecture. This prior also improves generalization, thus improving classification accuracy on CIFAR-10 [35] and ImageNet [14] datasets. APS does not require any additional learnable parameters, can be easily integrated into existing architectures, and also eliminates the need for overblurring the network's feature maps to improve classification consistency with respect to shifts.

2. Related work

Embedding invariance to shifts in the architecture of neural networks has been studied already in the 1980s [3, 38, 20, 39]. Robustness of neural networks to spatial and geometric transformations has been quantitatively assessed [33, 23, 18, 19, 2, 45, 43] and specialized neural network architectures have been proposed to improve invariance to transformations such as rotation, scaling and arbitrary deformations [10, 16, 56, 13, 54, 11, 53, 27, 51, 32]. Theoretical works based on wavelet filter banks [7, 46, 42] and multi-layer kernels [6, 5, 41] have addressed stability of neural networks to deformations. A body of work has also studied the robustness of CNNs to image corruptions [28, 52, 17, 21, 22]. Our focus is rather on restoring shift invariance lost due to subsampling.

A parallel line of work is adverserial training, whereby specifically designed perturbations with small norms are added to input images to yield large changes in the network's output [40, 50, 30, 24]. We focus on the robustness of networks to shifts which are examples of more naturally occurring transformations. Note that small shifts, despite being imperceptible, may yield images comparably far from the unshifted ones in l_p norms.

Data augmentation has been shown to improve robustness and generalization [39, 4, 25, 55, 15, 26, 12]. However, the robustness gains do not carry over to previously unseen transformations and out-of-dataset image distributions [22, 2]. Engstrom *et al.* [18] showed that while data augmentation improves robustness to shifts and rotations on average, there still exist transformations that can adversely impact the performance of the trained models.

Lack of shift invariance due to downsampling was noted early on by Simoncelli *et al.* [47]. They showed that shift invariance in multi-scale convolutional transforms is not possible, and instead proposed the notion of *shiftability*, a weaker form of invariance associated with anti-aliasing. Zhang [58] showed that both consistency to shifts and accuracy of classification can be improved by combining a dense max pooling layer with anti-aliasing before sampling. Zou *et al.* [59] used content-aware anti-aliasing to reduce risk of signal loss from over-blurring. Instead of explicitly anti-aliasing the feature maps, Sundaramoorthi and Wang [49] showed that by parameterizing convolutional filters with smooth Gauss–Hermite basis functions, CNN classifiers can attain translation insensitivity, a weak form of shift invariance. Azulay and Weiss [2], on the other hand, showed that while anti-aliasing can improve shift invariance, it offers only a partial solution. This is because the improved robustness to shifts is limited by the action of non-linear activations like ReLU, and does not generalize well to image patterns not seen during training.

Removing stride layers from CNNs can restore shift invariance [2]. This is indeed the case with networks based on *à trous* convolutions [57, 9, 8]. Alas, this leads to a dramatic increase in memory and computation requirements, rendering it an impractical strategy for large networks. Shift invariance in convolutional neural networks can also be lost because of boundary and padding effects that arise due to the finite support of input images [34, 31, 1]. This allows CNNs to encode absolute spatial locations in images.

3. Our proposed approach

3.1. Preliminaries

Shift invariance. An operation $\mathcal{G}$ is said to be shift invariant if for a signal x and its shifted version x_s, $\mathcal{G}(x) = \mathcal{G}(x_s)$. Similarly, it is shift equivariant if $\mathcal{G}(x_s) = (\mathcal{G}(x))_s$. Convolution is an example of a shift equivariant operator. We define $\mathcal{G}$ as sum-shift-invariant if $\sum \mathcal{G}(x_s) = \sum \mathcal{G}(x)$, where the summation is over the pixels of $\mathcal{G}(x)$ and $\mathcal{G}(x_s)$ respectively.

Convolutional neural networks for classification contain fully connected layers at the end which are not shift invariant. As a result, any shifts in convolutional feature maps of the final layer can impact the classifier's final output. Global average pooling, popularly used in CNN architectures like ResNet [25] and MobileNet [29] can solve this problem. These layers reduce the feature map of each channel in the final convolutional layer to a scalar by averaging. Thus, if the convolutional part of the network is sum-shift-invariant, the overall classifier architecture can be made shift invariant. Our analysis in subsequent sections assumes the use of global average pooling layers.

Polyphase components. For simplicity, we will consider downsampling of 1-D signals with stride 2. The analysis easily generalizes to images and volumes. Consider a 1-D signal $x_0(n)$ with discrete-time Fourier transform (DTFT) $X_0(e^{j\omega})$, which we will denote by $X_0(\omega)$ from hereon. The DTFT of a one-pixel-shifted version $x_1(n) = x_0(n-1)$ is given by $X_0(\omega)e^{-j\omega}$. Given $x_0(n)$, there are two ways to uniformly sample it with stride 2: we can choose to retain samples at either even or odd locations on the grid. These two possible downsampled outputs denoted by y_0 and y_1 are called the even and odd polyphase components of x_0, and can be expressed as $y_0(n) = x_0(2n)$ and $y_1(n) = x_0(2n-1)$. Notice that the even polyphase component of x_1 is the same as the odd counterpart of x_0 and vice versa. The downsampled outputs $y_0(n)$ and $y_1(n)$ have DTFTs given by

$$Y_0(\omega) = \frac{X_0(\omega/2) + X_0(\omega/2+\pi)}{2}, \tag{1}$$

$$Y_1(\omega) = \frac{(X_0(\omega/2) - X_0(\omega/2+\pi))e^{-j\omega/2}}{2}. \tag{2}$$

The terms in (1) and (2) corresponding to $(\omega/2+\pi)$ are called aliased components. They arise when x_0 contains high frequencies, and can cause significant degradation of the subsampled outputs. This is traditionally countered by anti-aliasing [44], a signal processing technique which removes high frequencies in x_0 by blurring before sampling.

Global average pooling operation on a signal $x_0(n)$ results in its mean and, ignoring a normalizing constant, is equal to $X_0(\omega = 0)$.

3.2. Key problem with downsampling

Downsampling is used in CNNs to increase the receptive field of convolutions, and to reduce the amount of memory and computation needed for training. With these goals, either of the two polyphase components of a 1-D signal is a 'valid' result of downsampling. However, when using conventional linear sampling, current neural network architectures always select the even component, rejecting the odd one. As a result, downsampling x_0 and its shifted version x_1 always results in different signals y_0 and y_1 which are highly unlikely to be equal [47] or sum-shift-invariant. Indeed, we can see from (1) and (2) that $Y_0(0) \neq Y_1(0)$. Anti-aliasing based methods [58] attempt to improve invariance by promoting similarity between y_0 and y_1. In particular, they restore sum-shift-invariance by blurring before downsampling, resulting in outputs y_0^a and y_1^a with DTFTs,

$$Y_0^a(\omega) = \frac{X_0(\omega/2)}{2},\; Y_1^a(\omega) = \frac{X_0(\omega/2)e^{-j\omega/2}}{2}. \tag{3}$$

The resulting y_0^a and y_1^a do satisfy $Y_0^a(0) = Y_1^a(0)$, i.e., $\sum y_0^a = \sum y_1^a$. This desirable equality, however,

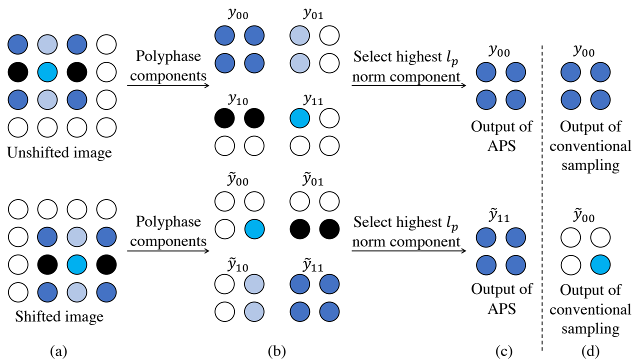

Figure 3. APS on single channel input. (a) Image and its shift. (b) The two images share the same set of polyphase components (with a potential shift between them). (c) By choosing the component with the highest l_p norm, APS returns the same output for both the images. (d) Output of conventional sampling in contrast.

is spoiled by the action of ReLU in subsequent layers. As Fig. 2(c) suggests, while y_0^a and y_1^a are similar, they are not identical. Minor differences between the signals become more prominent when they are thresholded by the ReLU non-linearity, resulting in $\sum \text{relu}(y_0^a) \neq \sum \text{relu}(y_1^a)$ (see A.2 in *supplementary material* for a more formal discussion). One could ask—can increasing the amount of blurring alleviate this problem caused by ReLUs? The answer is *no*. In fact, even ideal low-pass filtering does not help. This is because irrespective of the type of anti-aliasing used, y_0^a and y_1^a always have some differences and, therefore, are thresholded differently by ReLU.

While various non-linearities can spoil sum-shift-invariance similar to ReLU, exceptions like polynomial activations do not cause this problem. We state this formally in Theorem 1 (proof in A.1 in *supplementary material*).

Theorem 1. *Given non-linear activation function $g(y) = y^m$ with integer $m > 1$, and anti-aliased outputs of sampling y_0^a and y_1^a as defined above, we have*

$$\sum_{n \in \mathbb{Z}} g(y_0^a)(n) = \sum_{n \in \mathbb{Z}} g(y_1^a)(n). \tag{4}$$

Note that the above discussion and conclusions directly apply to 2-D images, with the difference that instead of 2, there exist 4 polyphase components to choose from.

3.3. Adaptive polyphase sampling

Consider stride-2 subsampling of a single channel image x. As shown in Fig. 3(a)-(b), the image can be downsampled along 4 possible grids, resulting in the set of 4 potential candidates for subsampling. We refer to these candidate results of sampling as polyphase components and denote them by $\{y_{ij}\}_{i,j=0}^1$. Similarly, the polyphase components of a 1-pixel shifted version of x, namely $\tilde{x} = x(m-1, n-1)$, are denoted by $\{\tilde{y}_{ij}\}_{i,j=0}^1$. Notice from Fig. 3(b) that $\{\tilde{y}_{ij}\}$ is just a re-ordered and potentially shifted version of the set $\{y_{ij}\}$. More formally,

$$\begin{aligned} \tilde{y}_{00} &= y_{11}(n_1-1, n_2-1), & \tilde{y}_{10} &= y_{01}(n_1, n_2-1), \\ \tilde{y}_{01} &= y_{10}(n_1-1, n_2), & \tilde{y}_{11} &= y_{00}(n_1, n_2). \end{aligned} \tag{5}$$

As we saw in Section 3.2, the key reason why conventional sampling is not shift invariant is that it always returns the first polyphase component of an image as output. This results in y_{00} and $\tilde{y}_{00}$ as subsampled outputs, which from (5) are not equal. To address this challenge, we propose adaptive polyphase sampling (APS). The key idea that APS exploits is that $\{y_{ij}\}$ and $\{\tilde{y}_{ij}\}$ are sets of identical[2] but re-ordered images. Therefore, the same subsampled output for x and $\tilde{x}$ can be obtained by selecting a polyphase component from $\{y_{ij}\}$ and $\{\tilde{y}_{ij}\}$ in a permutation invariant manner—for example choosing the one with the highest l_p norm. This is illustrated in Fig. 3(c). APS obtains its output y_{APS} by using the following criterion with $p = 2$.

$$\begin{gathered} y_{\text{APS}} = y_{i_1 j_1}, \\ \text{where } i_1, j_1 = \operatorname*{argmax}_{i,j} \{||y_{ij}||_p\}_{i,j=0}^1. \end{gathered} \tag{6}$$

For reference, conventional sampling returns $y_c = y_{00}$ as the output for x. It can be observed that for a shift $n_0 > 1$ between x and $\tilde{x}$, the resulting subsampled outputs y_{APS} and $\tilde{y}_{\text{APS}}$ are identical upto a shift $\sim \lceil \frac{n_0}{2} \rceil$, making the operation sum-shift-invariant. Additionally, note that since we did not use blurring in (6), y_{APS} will contain aliased components if x has high frequencies. This indicates that while anti-aliasing has been shown to improve shift invariance [58, 2], it is not strictly necessary.

When x is an image with C channels given by $x = (x_k)_{k=1}^C$, we define its polyphase components $\{y_{ij}\}_{i,j=0}^1$, by gathering the respective components for all channels, as shown in Fig. 4. In particular, if we assume each channel x_k, to have components $\{x_{k,ij}\}_{i,j=0}^1$, then for $i, j \in \{0, 1\}$,

$$y_{ij} = (x_{k,ij})_{k=1}^C. \tag{7}$$

The output of subsampling x using APS, denoted by y_{APS}, can then be obtained similar to (6). The above method can be extended to a general stride s, in a straightforward manner by norm maximization over s^2 polyphase components. The overall approach is summarized in Algorithm 1.

[2]The images in the two sets could have some shifts between them as well. However, this does not impact shift invariance for networks ending with global average pooling.

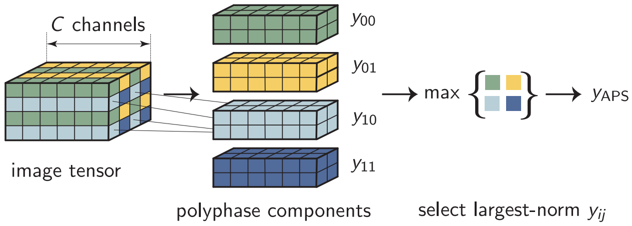

Figure 4. APS on multi-channel image tensor. The four polyphase components for a multi-channel input are constructed by gathering the components of the individual channels.

It is possible for APS to downsample inconsistently if two polyphase components in the set $\{y_{ij}\}_{i,j=0}^{1}$ have equal norms. However, in theory, assuming images to be drawn from a continuous distribution, this is an event with probability zero. It is also very less likely in practice based on our experiments. Note that it is for simplicity that APS chooses polyphase components using norm maximization. If needed, one could choose the components using more sophisticated permutation invariant ways as well.

3.3.1 Restoring shift invariance with APS

Downsampling an image and its shifted version with APS results in outputs that are exactly alike (up to a shift). Therefore, unlike the case with blurring, the action of *any* pointwise non-linearity on these outputs continues to yield identical signals. As a result, for a network containing convolutions, non-linear activations and APS layers, feature maps obtained from an image and its shift are either identical or shifted versions of each other, in all layers. When followed by global average pooling, this results in exactly identical outputs at the end and, therefore, *perfect shift invariance*.

3.4. Impact of boundary effects on shift invariance

While training CNNs to be shift invariant via data augmentation, a standard practice is to show the networks randomly shifted crops of images in the training set. The shifted images obtained this way have minor differences near their boundaries. After each layer these differences are amplified and propagated across the whole image to the point that shift invariance is lost even in the absence of downsampling. One way to resolve this problem is by padding images with enough zeros, though at the expense of additional computational and memory requirements.

To separate the two sources of loss in shift invariance—downsampling and boundary effects—while avoiding the extra overhead, we use circular padded convolutions and shifts in our experiments [58]. We show that with circular padding, CNNs with APS yield 100% classification consistency to shifts on CIFAR-10 and ImageNet datasets. We then train and evaluate the networks with standard padding and random crop based shifts as well, to still observe superior performance of APS over other approaches. Note that while circular padding helps reduce memory requirements, it can result in additional boundary artifacts when sampling odd sized images. However, even with these artifacts, APS results in better robustness to shifts as compared to prior methods (section E in *supplementary material*).

Algorithm 1 Adaptive Polyphase Sampling with stride s

1: **Input**: An image $x = \{x_k\}_{k=1}^{C}$ with C channels.
2: For $i, j \in \{0, 1, \dots s-1\}$, polyphase components:
$\{y_{ij}\} = x(sn_1 + i, sn_2 + j) = \{x_{k,ij}\}_{k=1}^{C}$.
3: **Output**: $y_{\text{APS}} = y_{i_1 j_1}$
where $i_1, j_1 = \text{argmax}_{i,j}\{||y_{ij}||_p\}_{i,j=0}^{s-1}$

3.5. Combining APS with anti-aliasing

We saw in Section 3.3 that APS can achieve perfect shift invariance without blurring the feature maps. While anti-aliasing is not strictly needed for shift invariance, it is still a useful tool to use before downsampling. This is because, as discussed in Section 3.1, it reduces signal degradation caused by aliased components during sampling. Hence, combining APS with anti-aliasing can help us in reaping the advantage of additional improvements in classification accuracy. This can be done by slightly blurring the feature maps before downsampling them with APS.

4. Experiments

We evaluate the performance of APS on CIFAR-10 [35] and ImageNet [14] datasets. For CIFAR-10, a 0.9/0.1 training/validation fractional split is used over the 50k training set with the final results reported over the test set of size 10k. For models trained on ImageNet, we report the best results evaluated on 50k validation set. APS is compared with 2 types of subsampling methods: (i) BlurPool (anti-aliased sampling) [58], and (ii) conventional subsampling, which we regard as baseline. We also evaluate models that combine APS and blurring. Filters of size 2×2, 3×3 and 5×5 have been used for anti-aliasing. The experiments are performed on different variants of the ResNet architecture [25] whose standard stride layers are replaced by the above subsampling modules. ResNet model embedded with filter size $j \times j$ is denoted by ResNet-LPFj. Similarly, ResNet-APSj denotes models containing APS and blur filter of size j. ResNet-18 is used as a running example in our experiments. We compare the models on four metrics.

- **Classification consistency on test set.** Likelihood of assigning an image and its shift to the same class.
- **Classification accuracy on unshifted test set.** The

Model	Consistency ResNet-20	ResNet-56	ResNet-18	ResNet-50	Accuracy (unshifted images) ResNet-20	ResNet-56	ResNet-18	ResNet-50
Baseline	90.83%	91.89%	90.88%	88.96%	89.76%	91.40%	91.96%	90.05%
APS	**100%**	**100%**	**100%**	**100%**	90.88%	92.66%	93.97%	94.05%
LPF-2	94.68%	94.44%	95.06%	92.47%	90.99%	92.07%	93.47%	91.61%
APS-2	**100%**	**100%**	**100%**	**100%**	91.69%	92.28%	94.38%	**94.27%**
LPF-3	95.23%	95.07%	97.19%	95.63%	91.01%	92.24%	94.01%	93.65%
APS-3	**100%**	**100%**	**100%**	**100%**	**91.78%**	92.72%	**94.53%**	93.80%
LPF-5	96.53%	96.90%	98.19%	97.38%	91.56%	**92.98%**	94.28%	94.12%
APS-5	**100%**	**100%**	**100%**	**100%**	91.75%	92.93%	94.48%	94.07%

Table 1. Classification consistency and accuracy (on unshifted images) evaluated on CIFAR-10 test set with ResNet models using blurring (LPF) and APS based downsampling. Circular padding was used in convolutional layers and circular shifts were used for consistency evaluation. The models were not shown any shifted images during training.

top-1 accuracy evaluated on unshifted images of the test dataset.

- **Invariance to out-of-distribution image patterns.** We measure classification consistency of the trained networks over image patterns not seen during training.
- **Stability of convolutional feature maps to small shifts.** We measure the extent to which a 1-pixel shift in input changes the feature maps inside convolutional neural networks. With y_l and $\tilde{y}_l$ as the feature maps for image x and its shift $\tilde{x}$ at depth l in a CNN, we use a shift compensated error $\delta(y_l, \tilde{y}_l)$ as the stability measure. It is defined as

$$\delta(y_l, \tilde{y}_l) = |\tilde{y}_l - T_j(y_l)|^2, \quad \text{where } j = \underset{j_1 \in S}{\operatorname{argmin}} \|\tilde{y}_l - T_{j_1}(y_l)\|_2, \tag{8}$$

$|\cdot|^2$ represents the squared magnitude function, T_j is an operator that shifts y_l by $j \in S$, and S is a set of all one pixel translations.

To separate the impact of boundary effects and downsampling on shift invariance, we first implement CNNs with circular padding and evaluate consistency on CIFAR-10 and ImageNet with random circular shifts up to 3 and 32 pixels respectively. All networks are trained with random horizontal flips and without any shifts unless mentioned otherwise. Models trained with random shifts are labelled DA. See *supplementary material* for more details on implementation.

4.1. Classification consistency and accuracy on test

We first evaluate 4 ResNet architectures, namely ResNet-20, 56, 18 and 50, with different downsampling modules on CIFAR-10 dataset. Originally used in [25] for CIFAR-10 classification, ResNet-20 and 56 are small models that downsample twice with stride 2 and contain

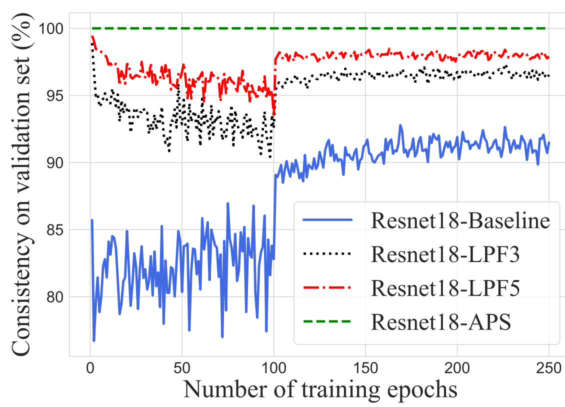

Figure 5. Classification consistency evaluated after each epoch on the CIFAR-10 validation set for 4 variants of Resnet-18: (i) Baseline, Blurpool with filter size (ii) 3 and (iii) 5, and (iv) APS.

$\{16, 32, 64\}$ filters in different layers. ResNet-18 and 50 on the other hand, downsample thrice with stride 2 and contain $\{64, 128, 256, 512\}$ filters. Table 1 shows consistency and accuracy of the models trained with circular padding. As expected, all networks containing APS modules exhibit perfect robustness to shifts evident from 100% classification consistency. Note that this is despite training the networks without showing any shifted versions of images. In contrast, the baseline ResNet-18 model is consistent 90.88% times, whereas its anti-aliased variants LPF-2, 3 and 5 show consistencies of 95.06%, 97.19%, 98.19% respectively. Similar to Zhang [58], we also observe increase in classification accuracy on unshifted test images with improving shift invariance. For instance, APS increases the accuracy of baseline ResNet-18 from 91.96% to 93.97%. We also observe that for a given blur filter size, accuracy obtained by combining APS and anti-aliasing is typically higher than the case with only blurring. As per Section 3.5, we believe this to be a result of combining the perfect shift invariance prior from APS, and anti-aliasing's

	Baseline	APS	LPF-2	APS-2	LPF-3	APS-3	LPF-5	APS-5
Consistency	80.39%	**100%**	84.35%	**100%**	86.54%	**99.996%**	87.88%	**99.98%**
Accuracy (unshifted images)	64.88%	67.05%	67.03%	**67.60%**	66.96%	67.43%	66.85%	67.52%

Table 2. Classification consistency and accuracy (on unshifted images) on ImageNet validation set obtained with ResNet-18 models containing circular padded convolutions and different subsampling modules. Networks were trained without any shifts during training.

	Consistency				Accuracy (unshifted images)			
Model	ResNet-20	ResNet-56	ResNet-18	ResNet-50	ResNet-20	ResNet-56	ResNet-18	ResNet-50
Baseline	88.89%	91.43%	90.81%	90.42%	90.06%	91.14%	91.60%	91.46%
APS	92.51%	94.04%	95.12%	95.21%	91.49%	92.89%	93.88%	93.63%
LPF-2	91.60%	92.30%	94.10%	91.81%	90.67%	91.80%	93.25%	91.93%
APS-2	93.62%	94.51%	95.82%	96.08%	91.44%	**93.04%**	93.63%	94.54%
LPF-3	91.84%	92.63%	94.82%	93.58%	91.73%	92.35%	94.15%	92.66%
APS-3	93.45%	94.59%	95.58%	**96.23%**	91.70%	93.02%	94.31%	**94.66%**
LPF-5	93.46%	93.53%	95.13%	94.70%	91.51%	92.55%	94.22%	93.61%
APS-5	**94.08%**	**94.61%**	**96.25%**	96.01%	**92.03%**	92.97%	**94.81%**	94.37%

Table 3. Classification consistency and accuracy (on unshifted images) obtained from ResNet models containing APS and blur based subsampling modules. Standard zero padded convolutions were used in the networks and random-crop based shifts were used for consistency evaluation. Networks were trained without any shifts during training.

ability to reduce signal degradation during sampling.

To understand the role of learned model weights on robustness to shifts, we compare how classification consistency on CIFAR-10 validation set varies while training ResNet-18 with the different sub-sampling methods. Fig. 5 shows that unlike the baseline and anti-aliased models, the validation consistency for APS is 100% throughout training. In fact, we observe perfect consistency in models with APS even before training, implying that APS truly embeds shift invariance into the CNN architecture.

We similarly compare the different downsampling methods on ImageNet with ResNet-18 architecture. The results are shown in Table 2. As expected, APS outperforms other methods with higher accuracy and near 100% consistency. Note that the minuscule fall in consistency for APS-3 and 5 occurred due to 2 and 10 images respectively (out of 50k) having polyphase components with the same l_2 norm. If needed, this rare occurrence can be avoided by using more robust polyphase component selection methods[3].

Boundary effects. As discussed in Section 3.4, evaluating consistency with random crop based shifts can result in shift invariance loss even in the absence of downsampling. Here, we investigate the impact of these boundary effects on the performance of APS. ResNet models with standard zero padding and different subsampling modules are trained and evaluated on CIFAR-10 dataset. For consistency evaluation, images are padded with zeros of size 3 on all sides, and a crop of size 32×32 is randomly chosen. Results in Table 3 reveal that for a given blur filter size, combining APS with anti-aliasing consistently provides better robustness and accuracy compared to blurring alone. In fact, in most cases, the consistency boost provided by APS with no anti-aliasing is still higher than the models that only use blurring.

4.2. Shift invariance on out-of-distribution images

Azulay and Weiss [2] showed that robustness to shifts achieved via data augmentation and anti-aliasing gets worse when the trained models are evaluated on images that differ substantially from the training distribution. In our experiments, we make a similar observation. On clean CIFAR-10 images, we train 4 variants of ResNet-18: model with (i) APS, (ii) LPF-5, (iii) APS + blur (APS-5), and (iv) model with vanilla subsampling but trained with random circular shifts of training set (referred to as DA). The trained networks are then evaluated for consistency on test images with small patches of pixels randomly erased from different locations. Fig. 6 shows that anti-aliased and data augmentation based models progressively lose robustness to shifts with increasing size of erased patches. On the other hand, models containing APS remain 100% shift invariant. In addition, the network with APS and blurring combined exhibits highest accuracy for all sizes of erasures.

We also evaluate these networks on a vertically flipped version of CIFAR-10 test set and observe similar results. Table 4 shows that, unlike data augmentation and blurring,

[3]For eg., it is much less likely for two images to have the same l_p and l_q norms for $p \neq q$. One could therefore maximize a sum of two norms to further reduce the likelihood of inconsistent sampling.

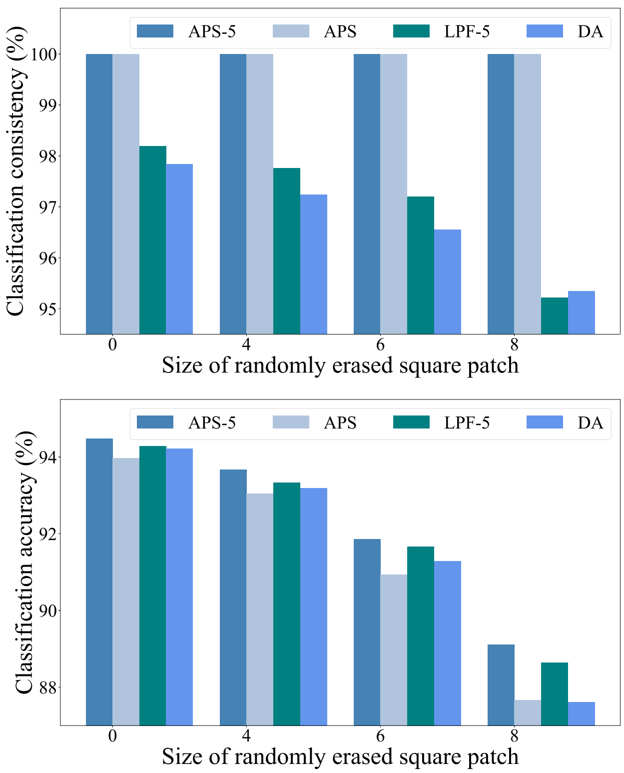

Figure 6. Classification consistency and accuracy on CIFAR-10 test images with randomly erased square patches. Unlike APS, models with data augmentation and blurring alone lose robustness to shifts as the images move away from training distribution.

	Consistency		Accuracy (unshifted)	
Models	Unflipped	Flipped	Unflipped	Flipped
APS-5	**100%**	**100%**	**94.48%**	**47.55%**
APS	**100%**	**100%**	93.97%	44.79%
LPF-5	98.19%	89.21%	94.28%	46.21%
DA	97.84%	84.94%	94.22%	44.97%

Table 4. Classification consistency and accuracy evaluated on vertically flipped CIFAR-10 test dataset. Unlike the case with data augmentation and blurring, models with APS continue to remain shift invariant on vertically flipping the dataset.

models based on APS continue to exhibit 100% consistency to shifts. Since, vertically flipping the dataset semantically pushes them further away from training set, all the models are expected to show poor accuracy. However, despite that, we observe APS-5 to show higher classification accuracy than other models.

4.3. Stability of internal convolutional feature maps to small shifts

We compare the impact of diagonally shifting an input image by 1-pixel on the feature maps of ResNet-18 models

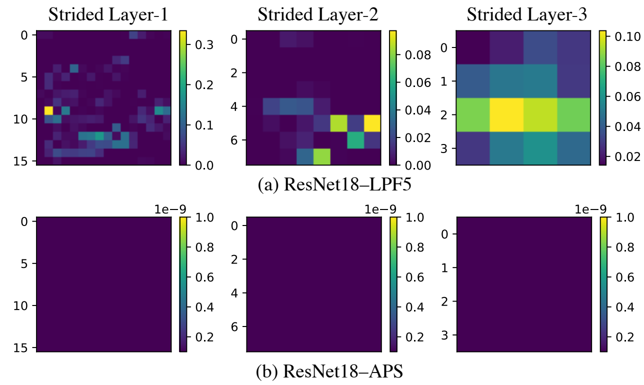

Figure 7. Stability of internal feature maps. A 1-pixel shift in input results in small differences in the feature maps for ResNet-18 with LPF-5 sampling. However, shift invariant nature of APS results in perfect stability in the feature maps of ResNet18-APS.

containing LPF-5 and APS modules. We compute feature maps for a CIFAR-10 test image and its shift, and compare them using shift compensated error δ from (8). Fig. 7 shows the errors for feature maps from the last 3 residual layers of the models (stride-2 sampling used in each layer). For each layer, we plot the errors for channels with the highest energy. The results indicate that while feature maps of LPF-5 model develop minor differences due to shift in input, the output of APS is completely stable.

5. Conclusion

Convolutional neural networks lose shift invariance due to subsampling (stride). We address this challenge by replacing the conventional linear sampling layers in CNNs with our proposed adaptive polyphase sampling (APS). A simple non-linear scheme, APS is the first approach that allows CNNs to be truly shift invariant. We show that with APS, the networks exhibit 100% consistency to shifts even *before training*. It also leads to better generalization performance, as evident from improved classification accuracy.

Acknowledgement

We thank Konik Kothari for useful discussions on the work presented in this paper. This research was supported by the European Research Council Starting Grant SWING, no. 852821. Numerical experiments were partly performed at sciCORE (http://scicore.unibas.ch/) scientific computing center at University of Basel. We also utilized resources supported by the National Science Foundation's Major Research Instrumentation program, grant #1725729, as well as the University of Illinois at Urbana-Champaign.

References

[1] Bilal Alsallakh, Narine Kokhlikyan, Vivek Miglani, Jun Yuan, and Orion Reblitz-Richardson. Mind the pad–cnns can develop blind spots. *arXiv preprint arXiv:2010.02178*, 2020.

[2] 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.

[3] Etienne Barnard and David Casasent. Shift invariance and the neocognitron. *Neural Networks*, 3(4):403 – 410, 1990.

[4] Yoshua Bengio, Frédéric Bastien, Arnaud Bergeron, Nicolas Boulanger–Lewandowski, Thomas Breuel, Youssouf Chherawala, Moustapha Cisse, Myriam Côté, Dumitru Erhan, Jeremy Eustache, Xavier Glorot, Xavier Muller, Sylvain Pannetier Lebeuf, Razvan Pascanu, Salah Rifai, François Savard, and Guillaume Sicard. Deep learners benefit more from out-of-distribution examples. volume 15 of *Proceedings of Machine Learning Research*, pages 164–172, Fort Lauderdale, FL, USA, 11–13 Apr 2011. JMLR Workshop and Conference Proceedings.

[5] Alberto Bietti and Julien Mairal. Invariance and stability of deep convolutional representations. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, pages 6210–6220. Curran Associates, Inc., 2017.

[6] Alberto Bietti and Julien Mairal. Group invariance, stability to deformations, and complexity of deep convolutional representations. *J. Mach. Learn. Res.*, 20(1):876–924, Jan. 2019.

[7] J. Bruna and S. Mallat. Invariant scattering convolution networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1872–1886, 2013.

[8] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018.

[9] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[10] G. Cheng, P. Zhou, and J. Han. Learning rotation-invariant convolutional neural networks for object detection in vhr optical remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, 54(12):7405–7415, 2016.

[11] Taco Cohen and Max Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999, 2016.

[12] Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[13] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

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

[16] Sander Dieleman, Jeffrey De Fauw, and Koray Kavukcuoglu. Exploiting cyclic symmetry in convolutional neural networks. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 1889–1898. JMLR.org, 2016.

[17] S. Dodge and L. Karam. A study and comparison of human and deep learning recognition performance under visual distortions. In *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–7, 2017.

[18] Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. Exploring the landscape of spatial robustness. volume 97 of *Proceedings of Machine Learning Research*, pages 1802–1811, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

[19] Alhussein Fawzi and Pascal Frossard. Manitest: Are classifiers really invariant? 2015.

[20] Kunihiko Fukushima. Neocognitron for handwritten digit recognition. *Neurocomputing*, 51:161 – 180, 2003.

[21] Robert Geirhos, David HJ Janssen, Heiko H Schütt, Jonas Rauber, Matthias Bethge, and Felix A Wichmann. Comparing deep neural networks against humans: object recognition when the signal gets weaker. *arXiv preprint arXiv:1706.06969*, 2017.

[22] Robert Geirhos, Carlos R. M. Temme, Jonas Rauber, Heiko H. Schütt, Matthias Bethge, and Felix A. Wichmann. Generalisation in humans and deep neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31, pages 7538–7550. Curran Associates, Inc., 2018.

[23] Ian Goodfellow, Honglak Lee, Quoc V. Le, Andrew Saxe, and Andrew Y. Ng. Measuring invariances in deep networks. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 646–654. Curran Associates, Inc., 2009.

[24] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[25] 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.

[26] Dan Hendrycks*, Norman Mu*, Ekin Dogus Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple method to improve robustness and uncertainty under data shift. In *International Conference on Learning Representations*, 2020.

[27] João F. Henriques and Andrea Vedaldi. Warped convolutions: Efficient invariance to spatial transformations. In

Proceedings of the 34th International Conference on Machine Learning - Volume 70, ICML'17, page 1461–1469. JMLR.org, 2017.

[28] H. Hosseini, B. Xiao, and R. Poovendran. Google's cloud vision api is not robust to noise. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 101–105, 2017.

[29] 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.

[30] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, pages 125–136, 2019.

[31] Md Amirul Islam*, Sen Jia*, and Neil D. B. Bruce. How much position information do convolutional neural networks encode? In *International Conference on Learning Representations*, 2020.

[32] Angjoo Kanazawa, Abhishek Sharma, and David W. Jacobs. Locally scale-invariant convolutional neural networks. *CoRR*, abs/1412.5104, 2014.

[33] Can Kanbak, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Geometric robustness of deep networks: Analysis and improvement. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[34] Osman Semih Kayhan and Jan C. van Gemert. On translation invariance in cnns: Convolutional layers can exploit absolute spatial location. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[35] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[36] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[37] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.

[38] Yann LeCun, Bernhard E. Boser, John S. Denker, Donnie Henderson, R. E. Howard, Wayne E. Hubbard, and Lawrence D. Jackel. Handwritten digit recognition with a back-propagation network. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, pages 396–404. Morgan-Kaufmann, 1990.

[39] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[40] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[41] Julien Mairal, Piotr Koniusz, Zaid Harchaoui, and Cordelia Schmid. Convolutional kernel networks. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27, pages 2627–2635. Curran Associates, Inc., 2014.

[42] Stéphane Mallat. Group invariant scattering. *Communications on Pure and Applied Mathematics*, 65(10):1331–1398, 2012.

[43] Marco Manfredi and Yu Wang. Shift equivariance in object detection. *arXiv preprint arXiv:2008.05787*, 2020.

[44] Alan V Oppenheim, John R Buck, and Ronald W Schafer. *Discrete-time signal processing. Vol. 2*. Upper Saddle River, NJ: Prentice Hall, 2001.

[45] Avraham Ruderman, Neil C Rabinowitz, Ari S Morcos, and Daniel Zoran. Pooling is neither necessary nor sufficient for appropriate deformation stability in cnns. *arXiv preprint arXiv:1804.04438*, 2018.

[46] Laurent Sifre and Stephane Mallat. Rotation, scaling and deformation invariant scattering for texture discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[47] Eero P Simoncelli, William T Freeman, Edward H Adelson, and David J Heeger. Shiftable multiscale transforms. *IEEE transactions on Information Theory*, 38(2):587–607, 1992.

[48] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[49] Ganesh Sundaramoorthi and Timothy E Wang. Translation insensitive cnns. *arXiv preprint arXiv:1911.11238*, 2019.

[50] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[51] Nanne van Noord and Eric Postma. Learning scale-variant and scale-invariant features for deep image classification. *Pattern Recognition*, 61:583 – 592, 2017.

[52] Igor Vasiljevic, Ayan Chakrabarti, and Gregory Shakhnarovich. Examining the impact of blur on recognition by convolutional networks. *arXiv preprint arXiv:1611.05760*, 2016.

[53] Maurice Weiler, Fred A. Hamprecht, and Martin Storath. Learning steerable filters for rotation equivariant cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[54] Daniel E. Worrall, Stephan J. Garbin, Daniyar Turmukhambetov, and Gabriel J. Brostow. Harmonic networks: Deep translation and rotation equivariance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[55] Ren Wu, Shengen Yan, Yi Shan, Qingqing Dang, and Gang Sun. Deep image: Scaling up image recognition. *arXiv preprint arXiv:1501.02876*, 7(8), 2015.

[56] Yichong Xu, Tianjun Xiao, Jiaxing Zhang, Kuiyuan Yang, and Zheng Zhang. Scale-invariant convolutional neural networks. *arXiv preprint arXiv:1411.6369*, 2014.

[57] Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. Dilated residual networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[58] Richard Zhang. Making convolutional networks shift-invariant again. volume 97 of *Proceedings of Machine Learning Research*, pages 7324–7334, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

[59] Xueyan Zou, Fanyi Xiao, Zhiding Yu, and Yong Jae Lee. Delving deeper into anti-aliasing in convnets. *arXiv preprint arXiv:2008.09604*, 2020.