# Making Convolutional Networks Shift-Invariant Again

**Richard Zhang** [1]

## Abstract

Modern convolutional networks are not shift-invariant, as small input shifts or translations can cause drastic changes in the output. Commonly used downsampling methods, such as max-pooling, strided-convolution, and average-pooling, ignore the sampling theorem. The well-known signal processing fix is anti-aliasing by low-pass filtering before downsampling. However, simply inserting this module into deep networks degrades performance; as a result, it is seldomly used today. We show that when integrated correctly, it is compatible with existing architectural components, such as max-pooling and strided-convolution. We observe *increased accuracy* in ImageNet classification, across several commonly-used architectures, such as ResNet, DenseNet, and MobileNet, indicating effective regularization. Furthermore, we observe *better generalization*, in terms of stability and robustness to input corruptions. Our results demonstrate that this classical signal processing technique has been undeservingly overlooked in modern deep networks.

## 1. Introduction

When downsampling a signal, such an image, the textbook solution is to anti-alias by low-pass filtering the signal (Oppenheim et al., 1999; Gonzalez & Woods, 1992). Without it, high-frequency components of the signal alias into lower-frequencies. This phenomenon is commonly illustrated in movies, where wheels appear to spin backwards, known as the Stroboscopic effect, due to the frame rate not meeting the classical sampling criterion (Nyquist, 1928). Interestingly, most modern convolutional networks do not worry about anti-aliasing.

Early networks did employ a form of blurred-downsampling – average pooling (LeCun et al., 1990). However, ample empirical evidence suggests max-pooling provides stronger task performance (Scherer et al., 2010), leading to its widespread adoption. Unfortunately, max-pooling does not provide the same anti-aliasing capability, and a curious, recently uncovered phenomenon emerges – small shifts in the input can drastically change the output (Engstrom et al., 2019; Azulay & Weiss, 2018). As seen in Figure 1, network outputs can oscillate depending on the input position.

Blurred-downsampling and max-pooling are commonly viewed as competing downsampling strategies (Scherer et al., 2010). However, we show that they are compatible. Our simple observation is that max-pooling is inherently composed of two operations: (1) evaluating the max operator densely and (2) naive subsampling. We propose to low-pass filter between them as a means of anti-aliasing. This viewpoint enables low-pass filtering to augment, rather than replace max-pooling. As a result, shifts in the input leave the output relatively unaffected (shift-invariance) and more closely shift the internal feature maps (shift-equivariance).

Furthermore, this enables proper placement of the low-pass filter, directly before subsampling. With this methodology, practical anti-aliasing can be achieved with any existing strided layer, such as strided-convolution, which is used in more modern networks such as ResNet (He et al., 2016) and MobileNet (Sandler et al., 2018).

A potential concern is that overaggressive filtering can result in heavy loss of information, degrading performance. However, we actually observe *increased accuracy* in ImageNet classification (Russakovsky et al., 2015) across architectures, as well as *increased robustness* and *stability* to corruptions and perturbations (Hendrycks et al., 2019). In summary:

- We integrate classic anti-aliasing to improve shift-equivariance of deep networks. Critically, the method is compatible with existing downsampling strategies.
- We validate on common downsampling strategies – max-pooling, average-pooling, strided-convolution – in different architectures. We test across multiple tasks – image classification and image-to-image translation.
- For ImageNet classification, we find, surprisingly, that accuracy increases, indicating effective regularization.
- Furthermore, we observe better generalization. Performance is more robust and stable to corruptions such as rotation, scaling, blurring, and noise variants.

[1]Adobe Research, San Francisco, CA. Correspondence to: Richard Zhang <rizhang@adobe.com>.

*Proceedings of the $36^{th}$ International Conference on Machine Learning*, Long Beach, California, PMLR 97, 2019. 

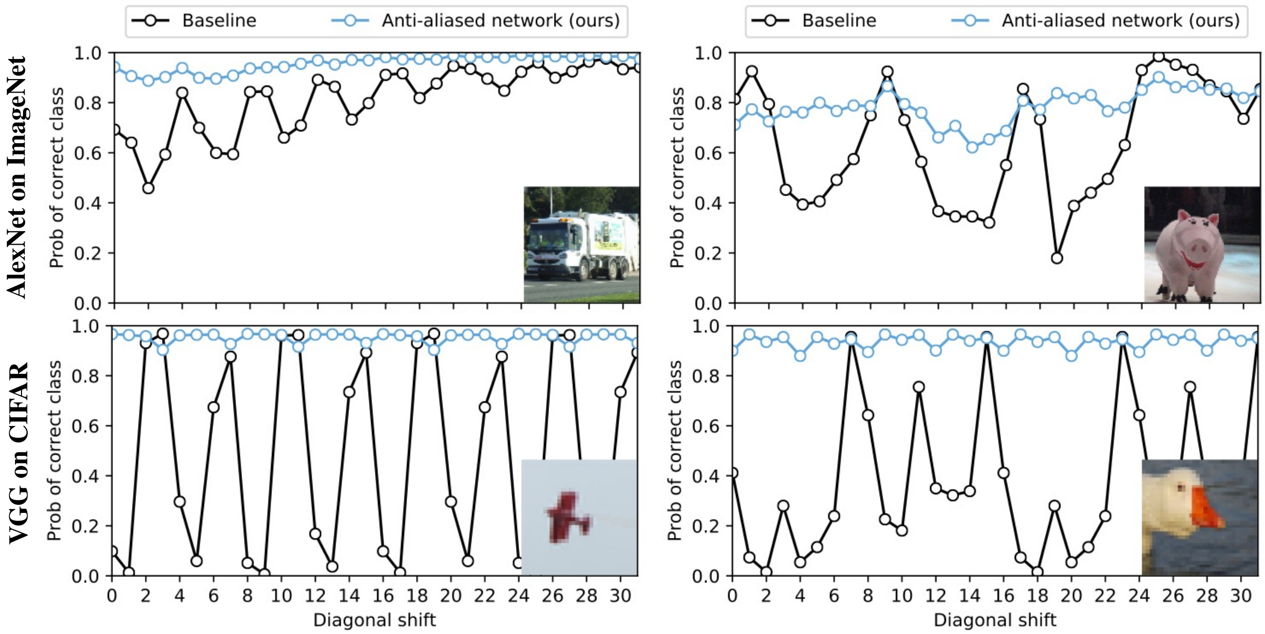



*Figure 1.* **Classification stability for selected images.** Predicted probability of the correct class changes when shifting the image. The baseline (black) exhibits chaotic behavior, which is stabilized by our method (blue). We find this behavior across networks and datasets. Here, we show selected examples using AlexNet on ImageNet **(top)** and VGG on CIFAR10 **(bottom)**. Code and anti-aliased versions of popular networks are available at `https://richzhang.github.io/antialiased-cnns/`.

## 2. Related Work

Local connectivity and weight sharing have been a central tenet of neural networks, including the Neocognitron (Fukushima & Miyake, 1982), LeNet (LeCun et al., 1998) and modern networks such as Alexnet (Krizhevsky et al., 2012), VGG (Simonyan & Zisserman, 2015), ResNet (He et al., 2016), and DenseNet (Huang et al., 2017). In biological systems, local connectivity was famously discovered in a cat's visual system (Hubel & Wiesel, 1962). Recent work has strived to add additional invariances, such as rotation, reflection, and scaling (Sifre & Mallat, 2013; Bruna & Mallat, 2013; Kanazawa et al., 2014; Cohen & Welling, 2016; Worrall et al., 2017; Esteves et al., 2018). We focus on shift-invariance, which is often taken for granted.

Though different properties have been engineered into networks, what factors and invariances does an emergent representation actually learn? Qualitative analysis of deep networks have included showing patches which activate hidden units (Girshick et al., 2014; Zhou et al., 2015), actively maximizing hidden units (Mordvintsev et al., 2015), and mapping features back into pixel space (Zeiler & Fergus, 2014; Hénaff & Simoncelli, 2016; Mahendran & Vedaldi, 2015; Dosovitskiy & Brox, 2016a;b; Nguyen et al., 2017). Our analysis is focused on a specific, low-level property and is complementary to these approaches.

A more quantitative approach for analyzing networks is measuring representation or output changes (or robustness to changes) in response to manually generated perturbations to the input, such as image transformations (Goodfellow et al., 2009; Lenc & Vedaldi, 2015; Azulay & Weiss, 2018), geometric transforms (Fawzi & Frossard, 2015; Ruderman et al., 2018), and CG renderings with various shape, poses, and colors (Aubry & Russell, 2015). A related line of work is adversarial examples, where input perturbations are purposely directed to produce large changes in the output. These perturbations can be on pixels (Goodfellow et al., 2014a;b), a single pixel (Su et al., 2019), small deformations (Xiao et al., 2018), or even affine transformations (Engstrom et al., 2019). We aim to make the network robust to the simplest of these types of attacks and perturbations: shifts. In doing so, we also observe increased robustness across other types of corruptions and perturbations (Hendrycks et al., 2019).

Classic hand-engineered computer vision and image processing representations, such as SIFT (Lowe, 1999), wavelets, and image pyramids (Adelson et al., 1984; Burt & Adelson, 1987) also extract features in a sliding window manner, often with some subsampling factor. As discussed in Simoncelli et al. (1992), literal shift-equivariance cannot hold when subsampling. Shift-equivariance can be recovered if features are extracted densely, for example textons (Leung & Malik, 2001), the Stationary Wavelet Transform (Fowler, 2005), and DenseSIFT (Vedaldi & Fulkerson, 2008). Deep networks can also be evaluated densely, by removing striding and making appropriate changes to subsequent layers by using *á trous*/dilated convolutions (Chen

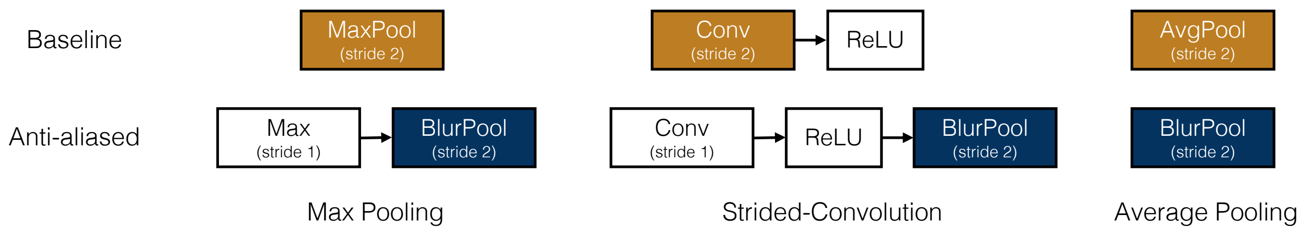



*Figure 2.* **Anti-aliasing common downsampling layers. (Top)** Max-pooling, strided-convolution, and average-pooling can each be better antialiased **(bottom)** with our proposed architectural modification. An example on max-pooling is shown below.

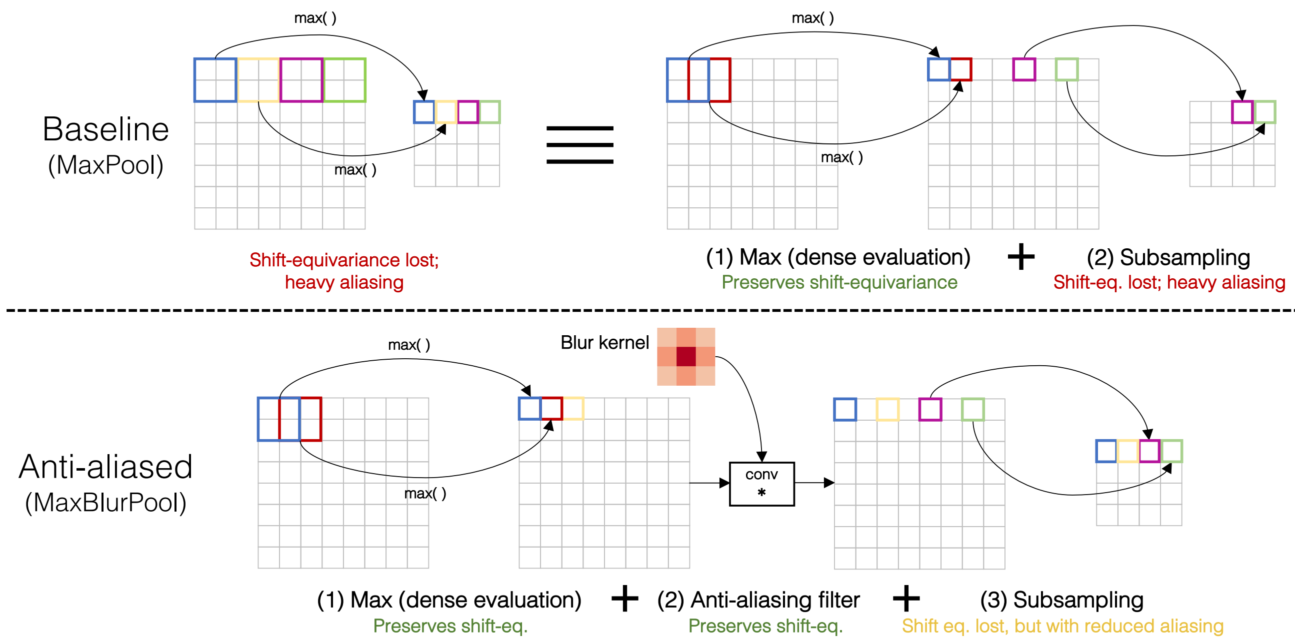



*Figure 3.* **Anti-aliased max-pooling. (Top)** Pooling does not preserve shift-equivariance. It is functionally equivalent to densely-evaluated pooling, followed by subsampling. The latter ignores the Nyquist sampling theorem and loses shift-equivariance. **(Bottom)** We low-pass filter between the operations. This keeps the first operation, while anti-aliasing the appropriate signal. Anti-aliasing and subsampling can be combined into one operation, which we refer to as **BlurPool**.

et al., 2015; 2018; Yu & Koltun, 2016; Yu et al., 2017). This comes at great computation and memory cost. Our work investigates improving shift-equivariance with minimal additional computation, by blurring before subsampling.

Early networks employed average pooling (LeCun et al., 1990), which is equivalent to blurred-downsampling with a box filter. However, work (Scherer et al., 2010) has found max-pooling to be more effective, which has consequently become the predominant method for downsampling. While previous work (Scherer et al., 2010; Hénaff & Simoncelli, 2016; Azulay & Weiss, 2018) acknowledges the drawbacks of max-pooling and benefits of blurred-downsampling, they are viewed as separate, discrete choices, preventing their combination. Interestingly, Lee et al. (2016) does not explore low-pass filters, but does propose to softly gate between max and average pooling. However, this does not fully utilize the anti-aliasing capability of average pooling.

Mairal et al. (2014) derive a network architecture, motivated by translation invariance, named Convolutional Kernel Networks. While theoretically interesting (Bietti & Mairal, 2017), CKNs perform at lower accuracy than contemporaries, resulting in limited usage. Interestingly, a byproduct of the derivation is a standard Gaussian filter; however, no guidance is provided on its proper integration with existing network components. Instead, we demonstrate practical integration with any strided layer, and empirically show performance increases on a challenging benchmark – ImageNet classification – on widely-used networks.

## 3. Methods

### 3.1. Preliminaries

**Deep convolutional networks as feature extractors** Let an image with resolution $H \times W$ be represented by $X \in \mathbb{R}^{H \times W \times 3}$. An $L$-layer CNN can be expressed

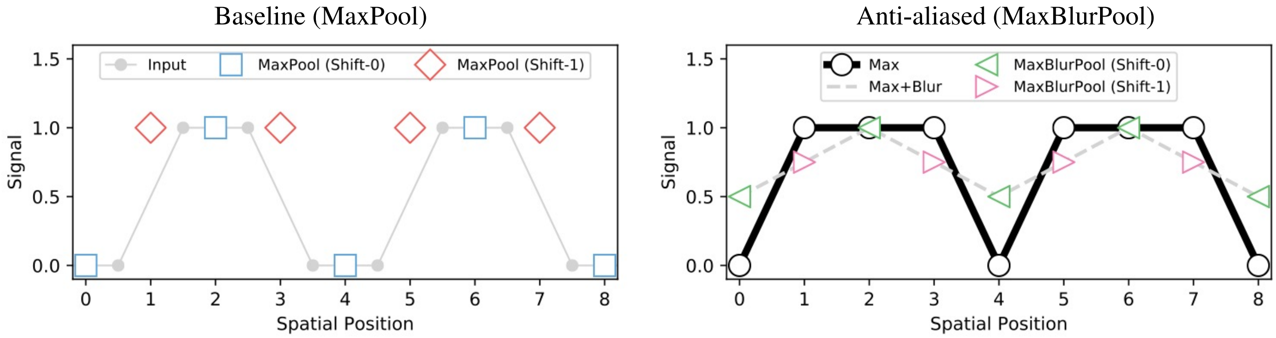



*Figure 4.* **Illustrative 1-D example of sensitivity to shifts.** We illustrate how downsampling affects shift-equivariance with a toy example. **(Left)** An input signal is in light gray line. Max-pooled ($k = 2$, $s = 2$) signal is in blue squares. Simply shifting the input and then max-pooling provides a completely different answer (red diamonds). **(Right)** The blue and red points are subsampled from a densely max-pooled ($k = 2$, $s = 1$) intermediate signal (**thick black line**). We low-pass filter this intermediate signal and then subsample from it, shown with green and magenta triangles, better preserving shift-equivariance.

as a feature extractor $\mathcal{F}_l(X) \in \mathbb{R}^{H_l \times W_l \times C_l}$, with layer $l \in \{0, 1, ..., L\}$, spatial resolution $H_l \times W_l$ and $C_l$ channels. Each feature map can also be upsampled to original resolution, $\widetilde{\mathcal{F}}_l(X) \in \mathbb{R}^{H \times W \times C_l}$.

**Shift-equivariance and invariance** A function $\widetilde{\mathcal{F}}$ is shift-equivariant if shifting the input equally shifts the output, meaning shifting and feature extraction are commutable.

$$\text{Shift}_{\Delta h, \Delta w}(\widetilde{\mathcal{F}}(X)) = \widetilde{\mathcal{F}}(\text{Shift}_{\Delta h, \Delta w}(X)) \quad \forall\, (\Delta h, \Delta w) \tag{1}$$

A representation is shift-invariant if shifting the input results in an *identical* representation.

$$\widetilde{\mathcal{F}}(X) = \widetilde{\mathcal{F}}(\text{Shift}_{\Delta h, \Delta w}(X)) \quad \forall\, (\Delta h, \Delta w) \tag{2}$$

**Periodic-N shift-equivariance/invariance** In some cases, the definitions in Eqns. 1, 2 may hold only when shifts $(\Delta h, \Delta w)$ are integer multiples of N. We refer to such scenarios as periodic shift-equivariance/invariance. For example, periodic-2 shift-invariance means that even-pixel shifts produce an identical output, but odd-pixel shifts may not.

**Circular convolution and shifting** Edge artifacts are an important consideration. When shifting, information is lost on one side and has to be filled in on the other.

In our CIFAR10 classification experiments, we use circular shifting and convolution. When the convolutional kernel hits the edge, it "rolls" to the other side. Similarly, when shifting, pixels are rolled off one edge to the other.

$$[\text{Shift}_{\Delta h, \Delta w}(X)]_{h,w,c} = X_{(h-\Delta h)\%H, (w-\Delta w)\%W, c}\,, \quad \text{where } \% \text{ is the modulus function} \tag{3}$$

The modification minorly affects performance and could be potentially mitigated by additional padding, at the expense of memory and computation. But importantly, this affords us a clean testbed. Any loss in shift-equivariance is purely due to characteristics of the feature extractor.

An alternative is to take a shifted crop from a larger image. We use this approach for ImageNet experiments, as it more closely matches standard train and test procedures.

### 3.2. Anti-aliasing to improve shift-equivariance

Conventional methods for reducing spatial resolution – max-pooling, average pooling, and strided convolution – all break shift-equivariance. We propose improvements, shown in Figure 2. We start by analyzing max-pooling.

**MaxPool→MaxBlurPool** Consider the example $[0, 0, 1, 1, 0, 0, 1, 1]$ signal in Figure 4 (left). Max-pooling (kernel k=2, stride s=2) will result in $[0, 1, 0, 1]$. Simply shifting the input results in a dramatically different answer of $[1, 1, 1, 1]$. Shift-equivariance is lost. These results are subsampling from an intermediate signal – the input densely max-pooled (stride-1), which we simply refer to as "max". As illustrated in Figure 3 (top), we can write max-pooling as a composition of two functions: $\text{MaxPool}_{k,s} = \text{Subsample}_s \circ \text{Max}_k$.

The Max operation preserves shift-equivariance, as it is densely evaluated in a sliding window fashion, but subsequent subsampling does not. We simply propose to add an anti-aliasing filter with kernel $m \times m$, denoted as $\text{Blur}_m$, as shown in Figure 4 (right). During implementation, blurring and subsampling are combined, as commonplace in image processing. We call this function $\text{BlurPool}_{m,s}$.

$$\begin{aligned} \text{MaxPool}_{k,s} \rightarrow\; & \text{Subsample}_s \circ \text{Blur}_m \circ \text{Max}_k \\ &= \text{BlurPool}_{m,s} \circ \text{Max}_k \end{aligned} \tag{4}$$

Sampling after low-pass filtering gives $[.5, 1, .5, 1]$ and $[.75, .75, .75, .75]$. These are closer to each other and better representations of the intermediate signal.

**StridedConv→ConvBlurPool** Strided-convolutions suffer from the same issue, and the same method applies.

$$\text{Relu} \circ \text{Conv}_{k,s} \rightarrow \text{BlurPool}_{m,s} \circ \text{Relu} \circ \text{Conv}_{k,1} \tag{5}$$

Importantly, this analogous modification applies conceptually to any strided layer, meaning the network designer can keep their original operation of choice.

**AveragePool→BlurPool** Blurred downsampling with a box filter is the same as average pooling. Replacing it with a stronger filter provides better shift-equivariance. We examine such filters next.

$$\text{AvgPool}_{k,s} \rightarrow \text{BlurPool}_{m,s} \tag{6}$$

**Anti-aliasing filter selection** The method allows for a choice of blur kernel. We test $m \times m$ filters ranging from size 2 to 5, with increasing smoothing. The weights are normalized. The filters are the outer product of the following vectors with themselves.

- ***Rectangle-2*** [1, 1]: moving average or box filter; equivalent to average pooling or "nearest" downsampling
- ***Triangle-3*** [1, 2, 1]: two box filters convolved together; equivalent to bilinear downsampling
- ***Binomial-5*** [1, 4, 6, 4, 1]: the box filter convolved with itself repeatedly; the standard filter used in Laplacian pyramids (Burt & Adelson, 1987)

# 4. Experiments

## 4.1. Testbeds

**CIFAR Classification** To begin, we test classification of low-resolution $32 \times 32$ images. The dataset contains 50k training and 10k validation images, classified into one of 10 categories. We dissect the VGG architecture (Simonyan & Zisserman, 2015), showing that shift-equivariance is a signal-processing property, progressively lost in each downsampling layer.

**ImageNet Classification** We then test on large-scale classification on $224 \times 224$ resolution images. The dataset contains 1.2M training and 50k validation images, classified into one of 1000 categories. We test across different architecture families – AlexNet (Krizhevsky & Hinton, 2009), VGG (Simonyan & Zisserman, 2015), ResNet (He et al., 2016), DenseNet (Huang et al., 2017), and MobileNet-v2 (Sandler et al., 2018) – with different downsampling strategies, as described in Table 1. Furthermore, we test the classifier robustness using the Imagenet-C and ImageNet-P datasets (Hendrycks et al., 2019).

**Conditional Image Generation** Finally, we show that the same aliasing issues in classification networks are also present in conditional image generation networks. We test on the Labels→Facades (Tyleček & Šára, 2013; Isola et al., 2017) dataset, where a network is tasked to generated a $256 \times 256$ photorealistic image from a label map. There are 400 training and 100 validation images.

| | ImageNet Classification | | | | | Generation |
|---|---|---|---|---|---|---|
| | Alex-Net | VGG | Res-Net | Dense-Net | Mobile-Netv2 | U-Net |
| **StridedConv** | $1^{\diamond}$ | – | $4^{\ddagger}$ | $1^{\ddagger}$ | $5^{\ddagger}$ | 8 |
| **MaxPool** | 3 | 5 | 1 | 1 | – | – |
| **AvgPool** | – | – | – | 3 | – | – |

*Table 1.* **Testbeds.** We test across tasks (ImageNet classification and Labels→Facades) and network architectures. Each architecture employs different downsampling strategies. We list how often each is used here. We can antialias each variant. $^{\diamond}$This convolution uses stride 4 (all others use 2). We only apply the antialiasing at stride 2. Evaluating the convolution at stride 1 would require large computation at full-resolution. $^{\ddagger}$For the same reason, we do not antialias the first strided-convolution in these networks.

## 4.2. Shift-Invariance/Equivariance Metrics

Ideally, a shift in the input would result in equally shifted feature maps internally:

**Internal feature distance.** We examine internal feature maps with $d(\text{Shift}_{\Delta h,\Delta w}(\widetilde{\mathcal{F}}(X)), \widetilde{\mathcal{F}}(\text{Shift}_{\Delta h,\Delta w}(X)))$ (left & right-hand sides of Eqn. 1). We use cosine distance, as common for deep features (Kiros et al., 2015; Zhang et al., 2018).

We can also measure the stability of the output:

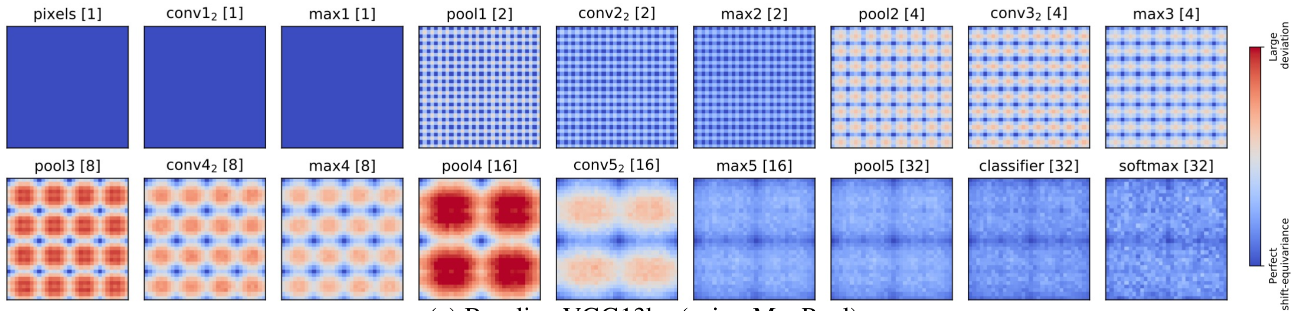
**Classification consistency.** For classification, we check how often the network outputs the same classification, given the same image with two different shifts: $\mathbb{E}_{X,h_1,w_1,h_2,w_2}\mathbb{1}\{\arg\max P(\text{Shift}_{h_1,w_1}(X)) = \arg\max P(\text{Shift}_{h_2,w_2}(X))\}$.

**Generation stability.** For image translation, we test if a shift in the input image generates a correspondingly shifted output. For simplicity, we test horizontal shifts. $\mathbb{E}_{X,\Delta w}\text{PSNR}\big(\text{Shift}_{0,\Delta w}(\mathcal{F}(X))), \mathcal{F}(\text{Shift}_{0,\Delta w}(X))\big)$.

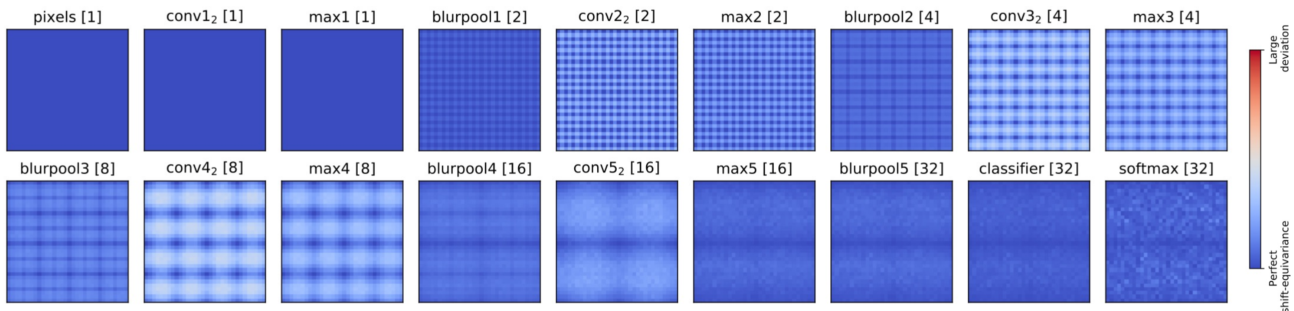
## 4.3. Internal shift-equivariance

We first test on the CIFAR dataset using the VGG13-bn (Simonyan & Zisserman, 2015) architecture.

We dissect the progressive loss of shift-equivariance by investigating the VGG architecture internally. The network contains 5 blocks of convolutions, each followed by max-pooling (with stride 2), followed by a linear classifier. For purposes of our understanding, MaxPool layers are broken into two components – before and after subsampling, e.g., `max1` and `pool1`, respectively. In Figure 5 (top), we show internal feature distance, as a function of all possible shift-offsets $(\Delta h, \Delta w)$ and layers. All layers before the first downsampling, `max1`, are shift-equivariant. Once downsampling occurs in `pool1`, shift-equivariance is lost. However, periodic-N shift-equivariance still holds, as indicated by the stippling pattern in `pool1`, and each subsequent subsampling doubles the factor N.




(a) Baseline VGG13bn (using MaxPool)




(b) Anti-aliased VGG13bn (using MaxBlurPool, *Bin-5*)

*Figure 5.* **Deviation from perfect shift-equivariance, throughout VGG.** Feature distance between left & right-hand sides of the shift-equivariance condition (Eqn 1). Each pixel in each heatmap is a shift $(\Delta h, \Delta w)$. Blue indicates perfect shift-equivariance; red indicates large deviation. Note that the dynamic ranges of distances are different per layer. For visualization, we calibrate by calculating the mean distance between two different images, and mapping red to half the value. Accumulated downsampling factor is in [brackets]; in layers `pool5`, `classifier`, and `softmax`, shift-equivariance and shift-invariance are equivalent, as features have no spatial extent. Layers up to `max1` have perfect equivariance, as no downsampling yet occurs. **(a)** On the **baseline network**, shift-equivariance is reduced each time downsampling takes place. Periodic-N shift-equivariance holds, with N doubling with each downsampling. **(b)** With our **antialiased network**, shift-equivariance is better maintained, and the resulting output is more shift-invariant.

In Figure 5 (bottom), we plot shift-equivariance maps with our anti-aliased network, using MaxBlurPool. Shift-equivariance is clearly better preserved. In particular, the severe drop-offs in downsampling layers do not occur. Improved shift-equivariance throughout the network cascades into more consistent classifications in the output, as shown by some selected examples in Figure 1. This study uses a *Bin-5* filter, trained without data augmentation. The trend holds for other filters and when training with augmentation.

### 4.4. Large-scale ImageNet classification

#### 4.4.1. Shift-invariance and accuracy

We next test on large-scale image classification of ImageNet (Russakovsky et al., 2015). In Figure 6, we show classification accuracy and consistency, across variants of several architectures – VGG, ResNet, DenseNet, and MobileNet-v2. The off-the-shelf networks are labeled as *Baseline*, and we use standard training schedules from the publicly available PyTorch (Paszke et al., 2017) repository for our anti-aliased networks. Each architecture has a different downsampling strategy, shown in Table 1. We typically refer to the popular ResNet50 as a running example; note that we see similar trends across network architectures.

**Improved shift-invariance** We apply progressively stronger filters – *Rect-2*, *Tri-3*, *Bin-5*. Doing so increases ResNet50 stability by $+0.8\%$, $+1.7\%$, and $+2.1\%$, respectively. Note that doubling layers – going to ResNet101 – only increases stability by $+0.6\%$. Even a simple, small low-pass filter, directly applied to ResNet50, outpaces this. As intended, stability increases across architectures (points move upwards in Figure 6).

**Improved classification** Filtering improves the shift-invariance. How does it affect absolute classification performance? We find that across the board, *performance actually increases* (points move to the right in Figure 6). The filters improve ResNet50 by $+0.7\%$ to $+0.9\%$. For reference, doubling the layers to ResNet101 increases accuracy by $+1.2\%$. A low-pass filter makes up much of this ground, without adding any learnable parameters. This is a surprising, unexpected result, as low-pass filtering removes information, and could be expected to reduce performance. On the contrary, we find that it serves as effective regularization, and these widely-used methods improve with simple anti-aliasing. As ImageNet-trained nets often serve as the backbone for downstream tuning, this improvement may be observed across other applications as well.

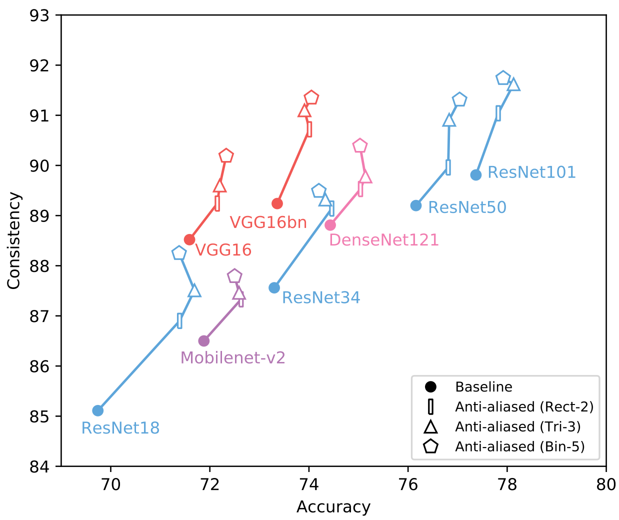



*Figure 6.* **ImageNet Classification consistency vs. accuracy.** Up (more consistent to shifts) and to the right (more accurate) is better. Different shapes correspond to the baseline (circle) or variants of our anti-aliased networks (bar, triangle, pentagon for length 2, 3, 5 filters, respectively). We test across network architectures. As expected, low-pass filtering helps shift-invariance. Surprisingly, classification accuracy is also improved.

The best performing filter varies by architecture, but all filters improve over the baseline. We recommend using the *Tri-3* or *Bin-5* filter. If shift-invariance is especially desired, stronger filters can be used.

#### 4.4.2. Out-of-distribution Robustness

We have shown increased stability (to shifts), as well as accuracy. Next, we test the generalization capability the classifier in these two aspects, using datasets from Hendrycks et al. (2019). We test stability to perturbations other than shifts. We then test accuracy on systematically corrupted images. Results are shown in Table 2, averaged across corruption types. We show the raw, unnormalized average, along with a weighted "normalized" average, as recommended.

**Stability to perturbations** The ImageNet-P dataset (Hendrycks et al., 2019) contains short video clips of a single image with small perturbations added, such as variants of noise (Gaussian and shot), blur (motion and zoom), simulated weather (snow and brightness), and geometric changes (rotation, scaling, and tilt). Stability is measured by flip rate (mFR) – how often the top-1 classification changes, on average, in consecutive frames. Baseline ResNet50 flips $7.9\%$ of the time; adding anti-aliasing *Bin-5* reduces by $1.0\%$. While antialiasing provides increased stability to shifts by design, a "free", emergent property is increased stability to other perturbation types.

**Robustness to corruptions** We observed increased accuracy on clean ImageNet. Here, we also observe more graceful degradation when images are corrupted. In addition to the previously explored corruptions, ImageNet-C contains impulse noise, defocus and glass blur, simulated frost and fog, and various digital alterations of contrast, elastic transformation, pixelation and jpeg compression. The geometric perturbations are not used. ResNet50 has mean error rate of $60.6\%$. Anti-aliasing with *Bin-5* reduces the error rate by $2.5\%$. As expected, the more "high-frequency" corruptions, such as adding noise and pixelation, show greater improvement. Interestingly, we see improvements even with "low-frequency" corruptions, such defocus blur and zoom blur operations as well.

| | Normalized average | | Unnormalized average | |
|---|---|---|---|---|
| | ImNet-C | ImNet-P | ImNet-C | ImNet-P |
| | mCE | mFR | mCE | mFR |
| **Baseline** | 76.4 | 58.0 | 60.6 | 7.92 |
| **Rect-2** | 75.2 | 56.3 | 59.5 | 7.71 |
| **Tri-3** | 73.7 | 51.9 | 58.4 | 7.05 |
| **Bin-5** | **73.4** | **51.2** | **58.1** | **6.90** |

*Table 2.* **Accuracy and stability robustness.** Accuracy in ImageNet-C, which contains systematically corrupted ImageNet images, measured by mean corruption error **mCE** (lower is better). Stability on ImageNet-P, which contains perturbed image sequences, measured by mean flip rate **mFR** (lower is better). We show raw, unnormalized scores, as well as scores normalized to AlexNet, as used in Hendrycks et al. (2019). Anti-aliasing improves both accuracy and stability over the baseline. All networks are variants of ResNet50.

Together, these results indicate that a byproduct of antialiasing is a more robust, generalizable network. Though motivated by shift-invariance, we actually observe increased stability to other perturbation types, as well as increased accuracy, both on clean and corrupted images.

### 4.5. Conditional image generation (Label→Facades)

We test on image generation, outputting an image of a facade given its semantic label map (Tyleček & Šára, 2013), in a GAN setup (Goodfellow et al., 2014a; Isola et al., 2017). Our classification experiments indicate that anti-aliasing is a natural choice for the discriminator, and is used in the recent StyleGAN method (Karras et al., 2019). Here, we explore its use in the generator, for the purposes of obtaining a shift-equivariant image-to-image translation network.

**Baseline** We use the pix2pix method (Isola et al., 2017). The method uses U-Net (Ronneberger et al., 2015), which contains 8 downsampling and 8 upsampling layers, with skip connections to preserve local information. No anti-aliasing filtering is applied in down or upsampling layers in the baseline. In Figure 7, we show a qualitative example, focusing in on a specific window. In the baseline (top), as the input $X$ shifts horizontally by $\Delta w$, the vertical bars on the generated window also shift. The generations start with

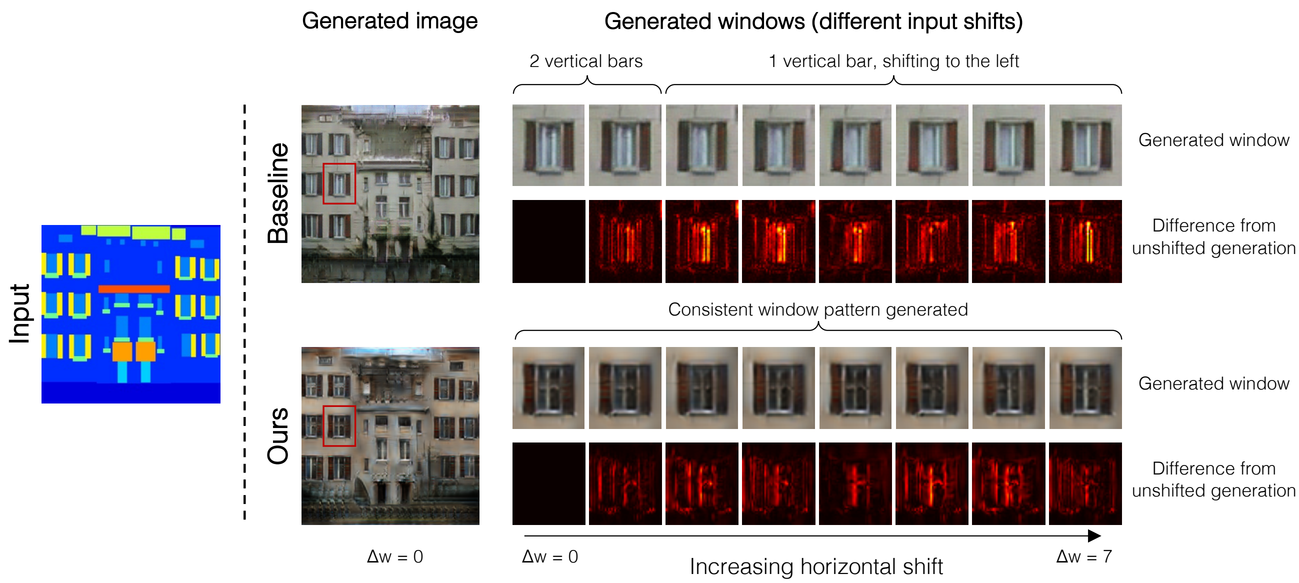



*Figure 7.* **Selected example of generation instability.** The left two images are generated facades from label maps. For the baseline method (top), input shifts cause different window patterns to emerge, due to naive downsampling and upsampling. Our method (bottom) stabilizes the output, generating the same window pattern, regardless the input shift.

| | Baseline | Rect-2 | Tri-3 | Bin-4 | Bin-5 |
|---|---|---|---|---|---|
| Stability [dB] | 29.0 | 30.1 | 30.8 | 31.2 | 34.4 |
| TV Norm $\times$100 | 7.48 | 7.07 | 6.25 | 5.84 | 6.28 |

*Table 3.* **Generation stability** PSNR (higher is better) between generated facades, given two horizontally shifted inputs. More aggressive filtering in the down and upsampling layers leads to a more shift-equivariant generator. **Total variation (TV) of generated images** (closer to ground truth images 7.80 is better). Increased filtering decreases the frequency content of generated images.

two bars, to a single bar, and eventually oscillates back to two bars. A shift-equivariant network would provide the same resulting facade, no matter the shift.

**Applying anti-aliasing** We augment the strided-convolution downsampling by blurring. The U-Net also uses upsampling layers, without any smoothing. Similar to the subsampling case, this leads to aliasing, in the form of grid artifacts (Odena et al., 2016). We mirror the downsampling by applying the same filter after upsampling. Note that applying the *Rect-2* and *Tri-3* filters while upsampling correspond to "nearest" and "bilinear" upsampling, respectively. By using the *Tri-3* filter, the same window pattern is generated, regardless of input shift, as seen in Figure 7 (bottom).

We measure similarity using peak signal-to-noise ratio between generated facades with shifted and non-shifted inputs: $\mathbb{E}_{X,\Delta w}\text{PSNR}(\text{Shift}_{0,\Delta w}(F(X)), F(\text{Shift}_{0,\Delta w}(X))))$. In Table 3, we show that the smoother the filter, the more shift-equivariant the output.

A concern with adding low-pass filtering is the loss of ability to generate high-frequency content, which is critical for generating high-quality imagery. Quantitatively, in Table 3, we compute the total variation (TV) norm of the generated images. Qualitatively, we observe that generation quality typically holds with the *Tri-3* filter and subsequently degrades. In the supplemental material, we show examples of applying increasingly aggressive filters. We observe a boost in shift-equivariance while maintaining generation quality, and then a tradeoff between the two factors.

These experiments demonstrate that the technique can make a drastically different architecture (U-Net) for a different task (generating pixels) more shift-equivariant.

## 5. Conclusions and Discussion

Shift-equivariance is lost in modern deep networks, as commonly used downsampling layers ignore Nyquist sampling and alias. We integrate low-pass filtering to anti-alias, a common signal processing technique. The simple modification achieves higher consistency, across architectures and downsampling techniques. In addition, in classification, we observe surprising boosts in accuracy and robustness.

Anti-aliasing for shift-equivariance is well-understood. A future direction is to better understand how it affects and improves generalization, as we observed empirically. Other directions include the potential benefit to downstream applications, such as nearest-neighbor retrieval, improving temporal consistency in video models, robustness to adversarial examples, and high-level vision tasks such as detection. Adding the inductive bias of shift-invariance serves as "built-in" shift-based data augmentation. This is potentially applicable to online learning scenarios, where the data distribution is changing.

ACKNOWLEDGMENTS

I am especially grateful to Eli Shechtman for helpful discussion and guidance. Michaël Gharbi, Andrew Owens, and anonymous reviewers provided beneficial feedback on earlier drafts. I thank labmates and mentors, past and present – Sylvain Paris, Oliver Wang, Alexei A. Efros, Angjoo Kanazawa, Taesung Park, and Phillip Isola – for their helpful comments and encouragement. I thank Dan Hendrycks for discussion about robustness tests on ImageNet-C/P.

## References

Adelson, E. H., Anderson, C. H., Bergen, J. R., Burt, P. J., and Ogden, J. M. Pyramid methods in image processing. *RCA engineer*, 29(6):33–41, 1984.

Aubry, M. and Russell, B. C. Understanding deep features with computer-generated imagery. In *ICCV*, 2015.

Azulay, A. and Weiss, Y. Why do deep convolutional networks generalize so poorly to small image transformations? In *arXiv*, 2018.

Bietti, A. and Mairal, J. Invariance and stability of deep convolutional representations. In *NIPS*, 2017.

Bruna, J. and Mallat, S. Invariant scattering convolution networks. *TPAMI*, 2013.

Burt, P. J. and Adelson, E. H. The laplacian pyramid as a compact image code. In *Readings in Computer Vision*, pp. 671–679. Elsevier, 1987.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *ICLR*, 2015.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 2018.

Cohen, T. and Welling, M. Group equivariant convolutional networks. In *ICML*, 2016.

Dosovitskiy, A. and Brox, T. Generating images with perceptual similarity metrics based on deep networks. In *NIPS*, 2016a.

Dosovitskiy, A. and Brox, T. Inverting visual representations with convolutional networks. In *CVPR*, 2016b.

Engstrom, L., Tsipras, D., Schmidt, L., and Madry, A. A rotation and a translation suffice: Fooling cnns with simple transformations. In *ICML*, 2019.

Esteves, C., Allen-Blanchette, C., Zhou, X., and Daniilidis, K. Polar transformer networks. In *ICLR*, 2018.

Fawzi, A. and Frossard, P. Manitest: Are classifiers really invariant? In *BMVC*, 2015.

Fowler, J. E. The redundant discrete wavelet transform and additive noise. *IEEE Signal Processing Letters*, 12(9): 629–632, 2005.

Fukushima, K. and Miyake, S. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pp. 267–285. Springer, 1982.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

Gonzalez, R. C. and Woods, R. E. *Digital Image Processing*. Pearson, 2nd edition, 1992.

Goodfellow, I., Lee, H., Le, Q. V., Saxe, A., and Ng, A. Y. Measuring invariances in deep networks. In *NIPS*, 2009.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *NIPS*, 2014a.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *ICLR*, 2014b.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, 2016.

Hénaff, O. J. and Simoncelli, E. P. Geodesics of learned representations. In *ICLR*, 2016.

Hendrycks, D., Lee, K., and Mazeika, M. Using pre-training can improve model robustness and uncertainty. In *ICLR*, 2019.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *CVPR*, 2017.

Hubel, D. H. and Wiesel, T. N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106–154, 1962.

Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017.

Kanazawa, A., Sharma, A., and Jacobs, D. Locally scale-invariant convolutional neural networks. In *NIPS Workshop*, 2014.

Karras, T., Laine, S., and Aila, T. A style-based generator architecture for generative adversarial networks. *ICLR*, 2019.

Kiros, R., Zhu, Y., Salakhutdinov, R. R., Zemel, R., Urtasun, R., Torralba, A., and Fidler, S. Skip-thought vectors. In *NIPS*, 2015.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E., and Jackel, L. D. Handwritten digit recognition with a back-propagation network. In *NIPS*, 1990.

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

Lee, C.-Y., Gallagher, P. W., and Tu, Z. Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree. In *AISTATS*, 2016.

Lenc, K. and Vedaldi, A. Understanding image representations by measuring their equivariance and equivalence. In *CVPR*, 2015.

Leung, T. and Malik, J. Representing and recognizing the visual appearance of materials using three-dimensional textons. *IJCV*, 2001.

Lowe, D. G. Object recognition from local scale-invariant features. In *ICCV*, 1999.

Mahendran, A. and Vedaldi, A. Understanding deep image representations by inverting them. In *CVPR*, 2015.

Mairal, J., Koniusz, P., Harchaoui, Z., and Schmid, C. Convolutional kernel networks. In *NIPS*, 2014.

Mordvintsev, A., Olah, C., and Tyka, M. Deepdream-a code example for visualizing neural networks. *Google Research*, 2:5, 2015.

Nguyen, A., Clune, J., Bengio, Y., Dosovitskiy, A., and Yosinski, J. Plug & play generative networks: Conditional iterative generation of images in latent space. In *CVPR*, 2017.

Nyquist, H. Certain topics in telegraph transmission theory. *Transactions of the American Institute of Electrical Engineers*, pp. 617–644, 1928.

Odena, A., Dumoulin, V., and Olah, C. Deconvolution and checkerboard artifacts. *Distill*, 2016. doi: 10.23915/distill.00003. URL http://distill.pub/2016/deconv-checkerboard.

Oppenheim, A. V., Schafer, R. W., and Buck, J. R. *Discrete-Time Signal Processing*. Pearson, 2nd edition, 1999.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in pytorch. 2017.

Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015.

Ruderman, A., Rabinowitz, N. C., Morcos, A. S., and Zoran, D. Pooling is neither necessary nor sufficient for appropriate deformation stability in cnns. In *arXiv*, 2018.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018.

Scherer, D., Muller, A., and Behnke, S. Evaluation of pooling operations in convolutional architectures for object recognition. In *ICANN*. 2010.

Sifre, L. and Mallat, S. Rotation, scaling and deformation invariant scattering for texture discrimination. In *CVPR*, 2013.

Simoncelli, E. P., Freeman, W. T., Adelson, E. H., and Heeger, D. J. Shiftable multiscale transforms. *IEEE transactions on Information Theory*, 38(2):587–607, 1992.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

Su, J., Vargas, D. V., and Sakurai, K. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 2019.

Tyleček, R. and Šára, R. Spatial pattern templates for recognition of objects with regular structure. In *German Conference on Pattern Recognition*, pp. 364–374. Springer, 2013.

Vedaldi, A. and Fulkerson, B. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008.

Worrall, D. E., Garbin, S. J., Turmukhambetov, D., and Brostow, G. J. Harmonic networks: Deep translation and rotation equivariance. In *CVPR*, 2017.

Xiao, C., Zhu, J.-Y., Li, B., He, W., Liu, M., and Song, D. Spatially transformed adversarial examples. *ICLR*, 2018.

Yu, F. and Koltun, V. Multi-scale context aggregation by dilated convolutions. *ICLR*, 2016.

Yu, F., Koltun, V., and Funkhouser, T. Dilated residual networks. In *CVPR*, 2017.

Zeiler, M. D. and Fergus, R. Visualizing and understanding convolutional networks. In *ECCV*, 2014.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. Object detectors emerge in deep scene cnns. In *ICLR*, 2015.