Spectral Leakage and Rethinking the Kernel Size in CNNs

Nergis Tomen, Jan C. van Gemert
Computer Vision Lab, Delft University of Technology
{n.tomen,j.c.vangemert}@tudelft.nl

Abstract

Convolutional layers in CNNs implement linear filters which decompose the input into different frequency bands. However, most modern architectures neglect standard principles of filter design when optimizing their model choices regarding the size and shape of the convolutional kernel. In this work, we consider the well-known problem of spectral leakage caused by windowing artifacts in filtering operations in the context of CNNs. We show that the small size of CNN kernels make them susceptible to spectral leakage, which may induce performance-degrading artifacts. To address this issue, we propose the use of larger kernel sizes along with the Hamming window function to alleviate leakage in CNN architectures. We demonstrate improved classification accuracy on multiple benchmark datasets including Fashion-MNIST, CIFAR-10, CIFAR-100 and ImageNet with the simple use of a standard window function in convolutional layers. Finally, we show that CNNs employing the Hamming window display increased robustness against various adversarial attacks. Our code is available online[1].

1. Introduction

A fundamental component in deep image recognition networks is the ability to non-linearly stack learned, shareable, linear mappings. The canonical example is the linear convolution operator in CNNs [15, 24, 47], while in recent visual Transformer models the query, key and values are token-shared linear mappings acting on pixel embeddings [6, 8]. These linear mappings, which in the visual domain typically take the form of filters, allow image feature learning. Such learnable, hierarchical, shareable, feature detectors are fundamental to the great success of deep learning [1, 5, 26], and a better understanding of these filters may broadly impact the whole field.

Image filters, such as local, oriented edge detectors, provide a highly reusable decomposition of the input image [7, 34, 46] and are accurate models of early biological vision [20]. From a deep, hierarchical feature learning perspective, it is interesting to ask how specialized reusable filters should be, and explore their response selectivity. In particular, here we investigate the role of frequency-selectivity of learned filters in deep networks.

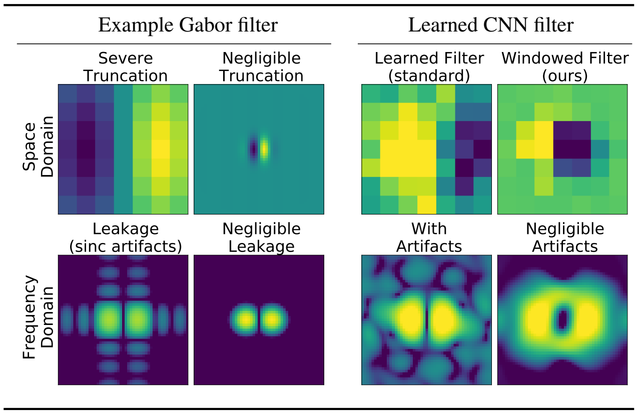


Figure 1. Windowing artifacts cause spectral leakage in the frequency domain, when a filter is not tapered off at the boundaries in the space domain. The top row shows example filters in space domain, while the bottom row are their corresponding frequency domains. Left: Example Gabor (bandpass) filter with severe truncation (kernel size 7×7) leads to spectral leakage in its frequency response due to sinc artifacts. The same filter with negligible truncation (kernel size 49×49) is a good quality bandpass filter. Right: A standard 7x7 CNN kernel trained on CIFAR-10 struggles to learn good quality bandpass filters, as the use of small kernel sizes typically lead to severe truncation. We propose using the standard Hamming window to taper off the kernels in space domain, which enables good quality bandpass frequency responses.

To investigate frequency-selectivity, we consider spectral leakage—which is a well-known [13, 35, 39] artifact in generic filtering operations—in the context of CNNs. The building block of CNNs, the convolution operator, can be thought of as a linear filter, which, due to the small kernel sizes employed in modern CNNs, is susceptible to spectral leakage. Although a well-studied concept in digital signal [35] and image processing [11, 19], we observe that a broader understanding of spectral leakage in deep networks has largely been neglected.

Spectral leakage, in the broad sense of the term, is when an operation on a signal introduces unwanted frequency

[1] https://github.com/ntomen/Windowed-Convolutions-for-CNNs

components to the result of that operation. In practice, the term leakage is typically used when a filter lets through frequency components of a signal outside of its intended passband due to windowing artifacts. For linear filters implemented via discrete convolution or cross-correlation operators, a kernel with finite size can be interpreted as a truncated version of an infinite, ideal filter. The finite size of the discrete kernel, within which the filter assumes non-zero values, represents a multiplication of an infinite kernel and a rectangular function in space domain, which translates as a convolution with a sinc function in frequency domain [39]. When a two-dimensional bandpass filter, such as a Gabor, is severely truncated, the rectangular function introduces windowing artifacts to the frequency response in the form of 'ripples' of the sinc function (Fig. 1, left).

Here we explore the effect of leakage artifacts in image classification performance in CNNs. We show that due to the typical choice of small kernel sizes, CNNs have little freedom to avoid rectangular truncation functions, which make them susceptible to spectral leakage (Fig. 1, right). We investigate the impact of leakage artifacts on benchmark classification tasks and demonstrate that the simple use of a standard window function which reduces leakage can improve classification accuracy. Furthermore, we show that windowed CNNs are more robust against certain types of adversarial examples.

Our contributions can be summarized as:

- We investigate the impact of spectral leakage in CNNs. Although spectral leakage is a fundamental concept in classical signal processing, its impact on CNN performance has not been explicitly explored before.
- We employ principles of good filter design, which are largely ignored in CNN models, to propose the use of larger kernels with the standard Hamming window function, which is tapered off at the kernel boundaries to alleviate spectral leakage.
- We demonstrate improvements to classification accuracy in benchmark datasets including Fashion-MNIST, CIFAR-10, CIFAR-100 and ImageNet with the simple use of a standard window function.
- We show that windowed CNNs display increased robustness against certain adversarial attacks including DeepFool and spatial transformation attacks.

2. Related work

2.1. Spectral leakage in signal processing

A signal observed within a finite window with aperiodic boundary conditions may be seen as a longer signal, truncated by multiplication with a rectangular window. This truncation introduces sidelobes, or 'ripples', to the spectral density of the signal and may, for example, decrease the signal-to-noise ratio in transmissions [35]. Often referred to as spectral leakage, such windowing artifacts are closely related to the Gibbs phenomenon [13] or ringing artifacts in digital image processing [11]. To combat the undesirable effects of leakage, window functions are commonly employed in many applications including spectral analysis via short-time Fourier transforms [39].

The use of 2-dimensional window functions is also commonplace for spectral decomposition in digital [19] and biomedical [41] image processing. Window functions are also an integral part of filter design, both in temporal [39] and image domains [3]. In this work, we consider the fundamental lack of window functions in CNN architectures and investigate whether leakage artifacts may lead to adverse effects. We propose the use of a Hamming window based on its reasonable attenuation of sidelobes while maintaining a relatively narrow main lobe.

2.2. Signal processing benefits for CNNs

Incorporating signal processing knowledge brought great benefits to deep learning. Marrying convolution to deep learning yields the CNN [25], the importance of which is difficult to overstate. The convolution theorem allows efficient CNN training [30]. The scattering transform [4] and its variants [36, 37, 45] allow encoding domain knowledge. Structuring filters based on scale-space theory [18, 43] brings data efficiency. Anti-aliasing in CNNs [57] increases robustness and accuracy. Reducing border effects in CNNs [21] improves translation equivariance and data efficiency. In this paper, we build on these successes and for the first time investigate spectral leakage in CNNs, where reducing spectral leakage increases classification accuracy and improves robustness.

2.3. Kernel size and shape in CNNs

In standard convolutional layers, using small kernel sizes reduces computational complexity and, empirically, improves accuracy, therefore increasingly small kernel sizes have been adopted over time in CNNs [15, 24, 47, 49, 50, 56]. One potential problem with larger kernel sizes may be over-parametrization, and our windowing method functions as a form of regularization which effectively constrains the parameter space. Unlike common regularization methods in deep learning, such as weight decay [14], dropout [48], early stopping [32] and data augmentation [44], our adopted Hamming window represents a spatially well-structured form of regularization which encourages a center-bias in the kernel shape. In fact, we show that our method is not a substitute for other types of regularization and synergizes well with weight decay and data augmentation.

Similarly, enforcing a center-bias in CNN filters is in line with the idea that images present hierarchically local problems [7, 34, 46], best tackled by local [28], deep, hierarchi-

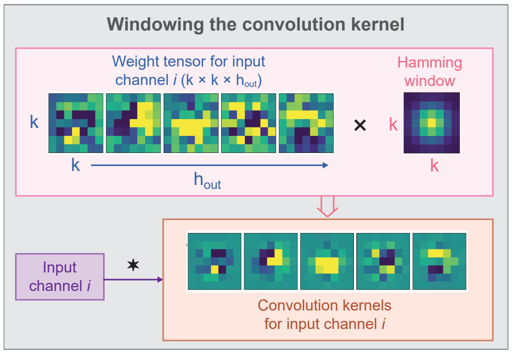

Windowing the convolution kernel
Weight tensor for input channel i (k × k × h_out)
Hamming window
k
k
h_out
×
k
k
Input channel i
∗
Convolution kernels for input channel i

Figure 2. Tapering the convolution kernels with the Hamming window. The typical weight tensor in a 2-D convolutional layer has size $(k \times k \times h_{in} \times h_{out})$. Here we show a single input channel i, which is convolved with h_{out} distinct $k \times k$ kernels, which are generated by multiplying each $k \times k$ slice of the weight tensor with the $k \times k$ Hamming window.

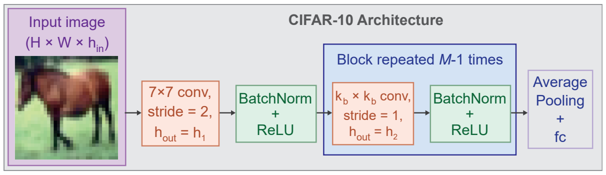


Figure 3. Simple architecture used for CIFAR-10, CIFAR-100, Fashion-MNIST and MNIST experiments. We vary the depth (number of convolutional layers M) of the network by repeating a convolution block (blue). The first layer downsamples the input via a strided convolution with a 7×7 kernel, similar to ResNet architectures, while the kernel size is k_b for all other convolutional layers. We also impose a channel bottleneck with the first layer having h_1 output channels whereas subsequent layers employ $h_2 > h_1$ output channels.

cal learning [1, 5, 26]. However, unlike previous work, we suggest that the use of small kernels may not be sufficient without explicit regularization of kernel boundaries.

Finally, other work has addressed structured kernel shapes in CNNs, using filter banks based on wavelets [4], Gabor filters [29], Gaussian derivatives [18] and circular harmonics [54]. Here, we focus explicitly on the truncation properties of the window function while learning the pixel weights in the conventional manner. This is similar to the approach of blurring the CNN filters [43], in our approach, however, we are not blurring the filters, or the feature maps [57]. Instead, we are performing a simple multiplication in space domain, unlike previous work.

2.4. Adversarial attacks

The features learned by deep models may not be robust which has implications for AI safety [2, 12]. An important observation in CNNs is that adversarial images with perturbations tiny in magnitude or imperceptible by humans [31, 51, 10, 42] can lead to misclassification with high confidence. Attacks can be generated based on information about the activations and gradients of the targeted network (white-box attacks) [12]. However, many adversarial examples are highly transferable between models [27] and without access to model parameters (black-box attacks), substitute models [38] or even simple spatial transformations [9] can be employed to generate adversarial images.

Adversarial examples can be traced back to brittle features which are inherent in the data distribution and are highly class-specific [16]. This means, that the reliance of classification models on very small magnitude features may boost performance, leading to a robustness-accuracy tradeoff [52]. Similarly, leakage artifacts are typically small in magnitude, but may be present in every feature map in a standard CNN with non-tapered filters. Here, we investigate whether simply using a tapered kernel may diminish filtering artifacts and provide both accuracy and robustness benefits over baseline models.

3. Convolution with the Hamming window

The discrete 2-dimensional convolution operation in CNNs can be described by

$$(f * g)[x_n, y_m] = \sum_{i=1}^{k} \sum_{j=1}^{k} g[x_i, y_j] \cdot f[x_{n-i}, y_{m-j}] \quad (1)$$

where $f[x_i, y_j]$ is the (padded) input image or feature map and $g[x_i, y_j]$ is a $k \times k$ kernel. From a discrete signal processing perspective, a linear filter can be implemented via a convolution. The aperiodic discrete convolution [35] between a signal f and an infinite kernel g' is given by

$$(f * g')[x_n, y_m] := \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} g'[x_i, y_j]\ f[x_{n-i}, y_{m-j}] \quad (2)$$

and this formulation can be used to describe an ideal, infinite impulse response (IIR) filter. In practice, to obtain kernels of finite size, in other words for finite impulse response (FIR) filter design [39], it is necessary to pick an appropriate window function $U[x_i, y_j]$ with $i, j \in \mathbb{Z}$ which limits the interval where the sum in Eq. 2 is non-zero. For CNNs, this window function is a rectangle function, equivalent to simple truncation. Formally, the rectangle window function

$$U[x_i, y_j] = \begin{cases} 1, & \text{if } 1 \leq i, j \leq k \\ 0, & \text{else} \end{cases} \quad (3)$$

multiplied with the ideal, infinite kernel g' in Eq. 2 as

$$\begin{aligned} &(f * g)[x_n, y_m] = \\ &\sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} g'[x_i, y_j]\, U[x_i, y_j]\ f[x_{n-i}, y_{m-j}] \end{aligned} \quad (4)$$

reduces to the CNN formulation in Eq. 1. Via the convolution theorem, multiplying g' with the rectangular function U in space domain corresponds to convolving the frequency response of g' with a sinc function in frequency domain.

Thus, windowing via simple truncation introduces potentially unwanted frequency components into the frequency response of the finite kernel $g[x, y]$, as shown in Fig. 1.

As an alternative to simple truncation, we propose to reduce unwanted frequency components through the standard Hamming window [13] in the convolution operations in a CNN. The one-dimensional Hamming window is a special case of the generalized cosine window, and is defined as

$$U[n] = \alpha - (1-\alpha) \cdot \cos\left(\frac{2\pi n}{N}\right), \quad 0 \le n \le N \qquad (5)$$

with $\alpha = 25/46$ [13, 39] and a window size of N discrete samples. We define the 2-D Hamming window as the outer product of two 1-D Hamming windows. The Hamming window can be implemented in standard architectures simply by multiplying each two-dimensional $k \times k$ kernel in a convolutional layer with the $k \times k$ Hamming window function (Fig. 2).

The Hamming window can be interpreted as a form of regularization. Multiplication with the window function reduces the gradient flow, or the effective learning rate, to the kernel boundaries which keeps the boundary weights close to zero and effectively shrinks the parameter space.

4. Experiments

4.1. Do CNN filters suffer from spectral leakage?

We devise a simple, fully controlled experiment to test whether the kernels in a single convolutional layer trained in a supervised setting display spectral leakage. To address this question, we force the network to learn good-quality bandpass filters in a regression task to predict the FFT magnitude of the input image. We create a synthetic dataset where the input images $S(x, y)$ are generated randomly via the superimposition of 2-D sine waves. Each input image

$$S(x, y) = \sum_{i=1}^{3} \sin(2\pi\, x_i' \,\omega_i + \phi_i), \qquad (6)$$

$$\text{with} \quad x_i' = x\cos(\theta_i) + y\sin(\theta_i) \qquad (7)$$

is the sum of three 2-D sine waves with spatial frequency ω_i sampled uniformly from $[0, 0.5\omega_s)$, where $0.5\omega_s$ is the Nyquist frequency. The orientation θ_i and phase ϕ_i of each sine wave are also sampled uniformly in the intervals $[0, \pi)$ and $[0, 2\pi)$ respectively. Each target (ground truth) vector is the flattened 2-D FFT magnitude of the corresponding input image. Input images are 32 × 32 pixels in size, hence the target values are vectors of length 1,024. Including the negative frequencies, the networks need to predict large values at 6 distinct frequency locations for each input image, as illustrated in Fig. 4.

We evaluate if a CNN can learn bandpass filters to approximate the discrete Fourier transform and use a single convolutional layer with 1,024 output channels, followed by ReLU and global average pooling. We test two CNN variants: one network with a Hamming window and one network without a Hamming window (baseline). To keep the central lobe size of the frequency responses similar between the two networks, we use a convolutional kernel size of k=7 for the baseline network and k=11 for the network with the Hamming window (see Supplement for a scan of kernel sizes). We train both networks using the mean squared error (MSE) and ADAM [22] optimizer on 10,000 training images and report the performance on an independent validation set of 1,000 images during training.

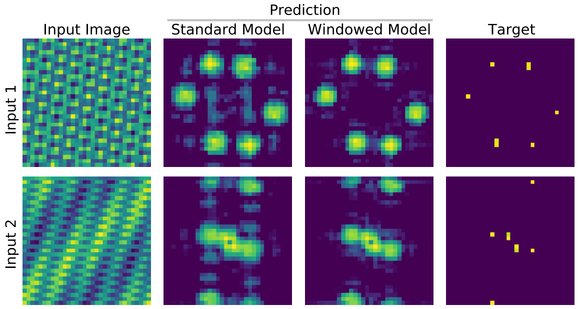


Figure 4. Learning to predict the FFT magnitude of an input image with a single convolutional layer. Network predictions for two example synthetic input images, randomly generated as the sum of three 2-D sine waves. The target vectors are the FFT magnitudes of the input images, including negative frequencies. We find that the model using Hamming windows alleviates leakage artifacts.

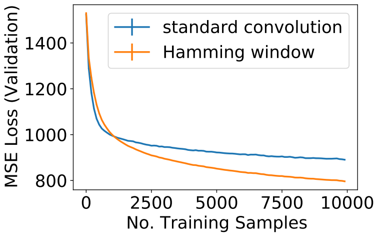


Figure 5. The windowed model achieves better regression performance on an independent validation set of 1000 images. The results are averaged over 5 runs with random model initializations (standard deviation error bars are too small to see).

Results in Fig. 5 show that longer training allows the windowed network to obtain a lower regression error on the validation set. This is partly due to the increased frequency resolution of the windowed network by using a larger kernel size, and partly due to artifact reduction. By visualizing the predictions of the trained networks in Fig. 4, we see that the bandpass filters learned by the baseline network, with standard convolutions, indeed suffers from leakage artifacts. In comparison, the windowed model is able to suppress responses which are adequately far from the target input frequencies.

This indicates that standard CNNs are susceptible to spectral leakage and will not readily learn filters which are tapered off at the boundaries in the absence of explicit reg-

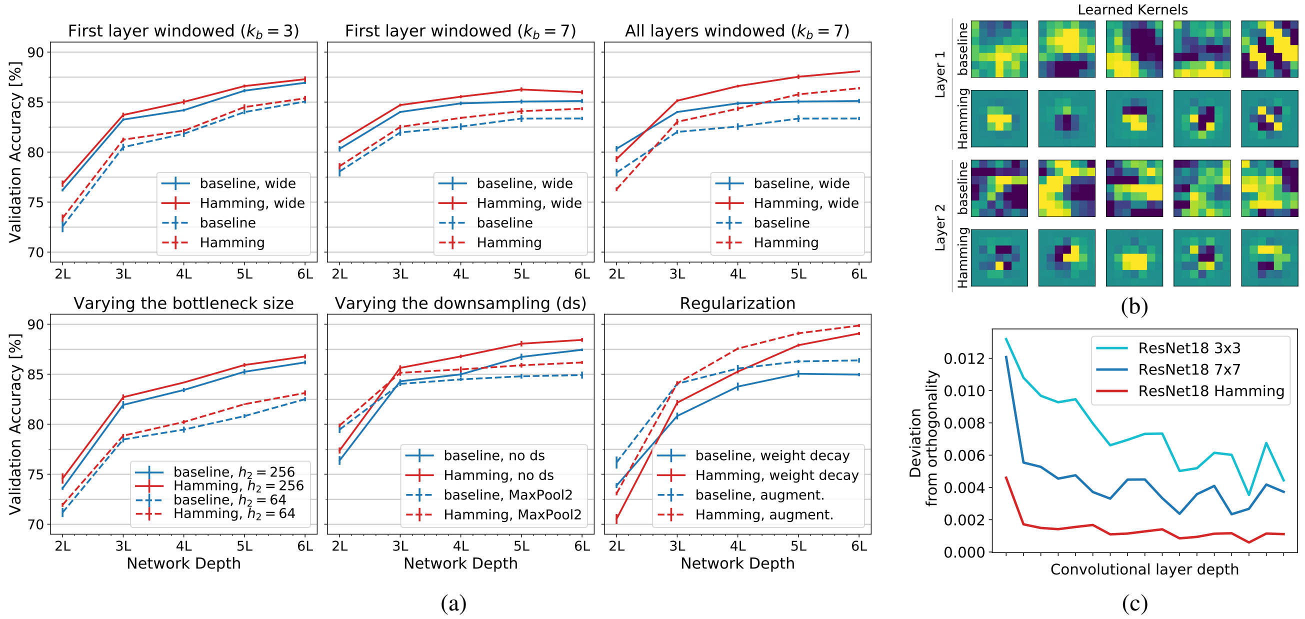


Figure 6. (a) CIFAR-10 validation accuracy as a function of varying network depth ($M = 2 \dots 6$ convolutional layers), in models using the Hamming window (red) and baselines with standard convolutional layers (blue). Line plots depict the average of 5 runs with error bars denoting standard deviation. We find that for all architecture variants we tried (Hamming window only on the first layer, Hamming window on all layers, different channel width, different methods of downsampling and regularization), models using the Hamming window consistently outperform the baseline models in networks deeper than 2 layers. (b) Example kernels after training in the network variant with $k_b = 7$ where the Hamming window is applied to all layers. (c) For ResNet18 models trained on ImageNet, we find that the deviation of each convolutional layer from a row orthogonal convolution [53] is lowest for the Hamming model compared to baselines.

ularization, even when leakage artifacts directly contribute to the loss. We find that a standard Hamming window can be employed to regularize the kernel weights and combat leakage artifacts. However, it is not clear from this toy experiment whether deeper networks with a large number of non-linearities can learn to suppress performance-degrading artifacts. Therefore, we investigate the effects of windowing in deeper networks next.

4.2. When does spectral leakage hurt classification?

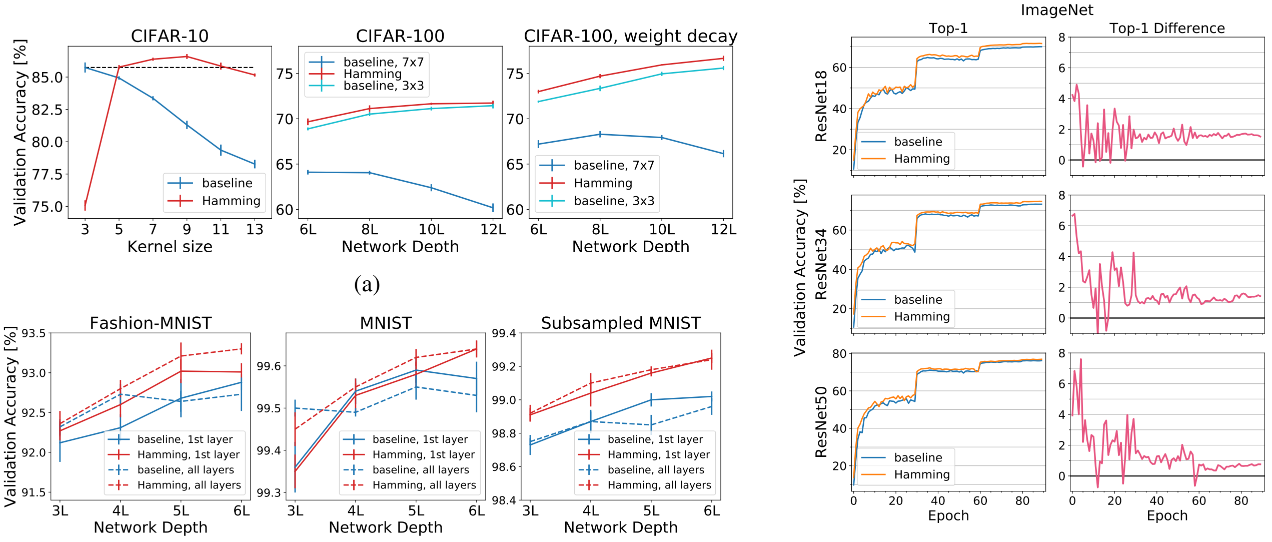
We extensively evaluate on CIFAR-10 and CIFAR-100 with model variations based on Fig. 3.

CIFAR-10. For all experiments, we train for 50 epochs using cross-entropy loss and SGD with a mini-batch size of 32 and momentum 0.9. The initial learning rate is 0.01 and decays by a factor of 0.1 at epochs 25 and 40. We vary network width and depth where we evaluate from 2 layers deep up to 6 layers deep. Unless stated otherwise, the number of output channels in the convolutional layers are $h_1 = 32$ and $h_2 = 128$ for the original networks and $h_1 = 64$ and $h_2 = 256$ for 'wide' networks. The windowed and baseline networks are trained identically and repeated 5 times with different random seeds.

First layer. The earlier layers may provide sufficiently powerful and shareable features for deeper layers. Thus, we evaluate a Hamming window in only the first convolutional layer. We test networks where the deeper layers have a kernel size of 3×3 (k_b=3) and 7×7 (k_b=7) and find that the accuracy of windowed models is consistently higher than the baseline. This is true both for the original and wide models (Fig. 6a, top row). Note that the performance increase is relatively small, it is caused by a Hamming window *only* in the first convolutional layer, while the rest of the architecture and hyperparameters are identical.

All layers. Next, we test whether alleviating spectral leakage in deeper layers. As windowing very small kernels is not meaningful, and we would like to keep the number of parameters in the baseline and windowed networks the same, we use a kernel size of 7×7 (k_b=7) in all layers. Notably, we find that using the Hamming window in all convolutional layers provides a significant boost to CIFAR-10 validation accuracy, especially in deeper networks (Fig. 6a, top row, right). To illustrate the learned weights, some example kernels from trained networks are shown in Fig. 6b.

Feature sharing. We hypothesise that artifact-free band-pass filters offers better shareable representations. Thus, the channel-wise network bottlenecks, which forces stronger feature sharing, may effect performance. To test this, we vary the bottleneck size in the original network ($h_1 = 32$) by changing the number of output channels in the deeper layers to $h_2 = 64$ and $h_2 = 256$, while using a Hamming window only in the first layer. Interestingly, we find that using a larger or smaller bottleneck size does not seem to affect the accuracy increase provided by the windowed con-



(a)

(b)

(c)

Figure 7. (a) Left: CIFAR-10 validation accuracy decreases monotonically with increasing kernel size for the baseline model, while a larger 9×9 kernel size maximizes performance for the model with the Hamming window. Middle and Right: 'Hamming' models with 7×7 kernels (red) outperform baseline networks with both 7×7 (blue) and 3×3 kernels (cyan) on the CIFAR-100 dataset. The performance boost is more pronounced when the networks are regularized by weight decay. (See Supplement C for weight decay sweeps.) (b) Left and middle: Fashion-MNIST and MNIST validation accuracy as a function of varying network depth. Right: We find that the benefits of windowing are more pronounced when the magnitude of high frequency components is increased by subsampling the input images. (c) Left: ImageNet validation accuracy is higher for ResNet architectures with the Hamming window than baseline ResNet models throughout training. Right: The difference in ImageNet validation accuracy between the Hamming and baseline ResNet models during training. (Also see Supplement D for training and validation loss.)

volution (Fig. 6a, bottom row, left).

Aliasing. Downsampling layers in CNNs are known to introduce performance-degrading aliasing artifacts [57]. We investigate whether the Hamming window may also be indirectly suppressing task-irrelevant, aliased frequency components. Thus, we train networks with no downsampling layers, and replace the strided convolution with a standard convolution (stride=1) while windowing only the first layer. As another control experiment, we also train networks which perform downsampling via a max-pooling layer with a 2×2 window instead of a strided convolution. We find that in both cases using a Hamming window still improves CIFAR-10 validation accuracy (Fig. 6a, bottom row, middle), which indicates that the performance increase provided by windowed convolutions is independent of aliasing and the choice of downsampling method.

Regularization. Our windowing regularizes the kernel weights close to the boundaries (Fig. 6b). We compare this regularization with other common methods, namely weight decay and data augmentation. We train networks with a weight decay value of 0.001, and independently, we train networks with random translation and horizontal flip augmentations. We find that, in explicitly regularized networks, not only do the benefits of our windowing not disappear, but the accuracy boost is in fact larger, especially for deeper networks (Fig. 6a, bottom row, right).

Optimal kernel size. Standard convolutional layers typically employ very small (3×3) kernel sizes. Although we only used a kernel size of 7×7 so far for our proposed method, it is not clear *a priori* what kernel size would be optimal for the Hamming window. To test this empirically, we vary the kernel size in all layers of a $M{=}6$ layer network, with or without the Hamming window. We find that while classification performance decreases monotonically with increasing kernel size beyond 3×3 for the baseline network, there is a larger, optimal kernel size which maximizes performance for the network using the Hamming window (Fig. 7a, left). In fact, we find that windowed networks with both kernel sizes $k{=}7$ and $k{=}9$ provide a significant accuracy improvement (outside of the standard deviation) over the best baseline model with $k{=}3$. We have also run experiments with Hann and Blackman windows (not shown) and found no difference to Hamming window.

CIFAR-100. For CIFAR-100 [23] experiments we employ wider models with $h_1{=}128$ and $h_2{=}256$, deeper models with up to 12 layers, and for the 'Hamming' models we use windowed convolutions in all layers. We train all models for 150 epochs, with initial learning rate 0.01 decaying by a factor of 0.1 at epochs 75 and 120. We also employ standard data augmentation (horizontal flip and random translations). Otherwise, the hyperparameters are the same as in the CIFAR-10 experiments. As an additional control, we run baselines with 7×7 kernel size (same number of parameters as the ‘Hamming’ model) and 3×3 kernel size (best baseline performance) in all layers. We find that windowed networks perform consistently better than both baselines (Fig. 7a, middle). The accuracy enhancement provided by the Hamming window is more pronounced with a weight decay of 0.001 (Fig. 7a, right).

4.3. Datasets with limited frequencies

Natural images may contain class-specific information in all frequency bands, which means spectral leakage between different frequency components may hinder discrimination of class-specific responses. We hypothesize that for less natural images, where not all frequency components are well-represented in the training set, the effects of windowing would be less prominent. To test this, we evaluate classification performance on Fashion-MNIST [55] and MNIST [25] datasets. Training parameters in this section are identical to the CIFAR-10 experiments. We train two types of models: one with convolutions with the Hamming window only in the first layer and $k_b = 3$ and one with Hamming window in all layers with $k_b = 7$.

For the Fashion-MNIST dataset, we find that the use of a Hamming window persistently improves classification performance (Fig. 7b, left), however the increase in validation accuracy is smaller than it is with more natural images found in CIFAR-10 and CIFAR-100 datasets. For the MNIST dataset, we don’t observe a performance increase in ‘Hamming’ models for most networks, and only a modest one for deeper networks (Fig. 7b, middle).

We attribute the lack of benefits from windowed convolutions in the MNIST dataset, to some degree, to the lack of high frequency components, whereby leakage in lowpass and bandpass filters cannot contaminate high frequency information. To test this, we subsample the 28×28 input images in the MNIST dataset, via bilinear interpolation, down to 14×14 images. Subsampling has the effect of increasing the relative magnitude of high frequency components, and we find that the Hamming window provides significant accuracy improvements in the subsampled MNIST (Fig. 7b, right). In particular, we find that both ‘Hamming’ models (windowing only the first layer or all layers) perform better than both baselines (including when $k_b = 3$).

4.4. ImageNet

We train ResNet [15] and VGG [47] models on the ImageNet [40] dataset for 90 epochs, with initial learning rate of 0.1 decaying by 0.1 at epochs 30 and 60. Optimization is performed using SGD with momentum 0.9 and weight decay 10^{-4}. Input images are randomly resized and cropped to 224×224 pixels and horizontally flipped. Baseline networks are VGG architectures with batch normalization [17] and kernel size $k = 3$ or $k = 7$, and the standard ResNet architectures [15] with $k = 7$ in the first layer, and $k = 3$ or $k = 7$ in all deeper layers. For the windowed networks, we replace all convolutions with Hamming-windowed convolutions with $k = 7$.

We find that when enforcing better frequency-selectivity, ImageNet validation accuracy is higher than baselines throughout training (Fig. 7c). This indicates that using standard window functions as an inductive prior helps the network settle early on to solutions which generalize better. Overall, we find that replacing the convolutional layers with Hamming-windowed layers provides accuracy improvements on the ImageNet benchmark (Fig. 8, Table 1).

Orthogonality. Spectral leakage may render CNNs unable to learn filters with non-overlapping frequency responses, thus leading to redundant representations. Similar to our windowed layers, redundancy reduction and performance increase can also be achieved by orthogonal convolutions [53]. Therefore, we analyse the effects of the Hamming window on the orthogonality of the weights learned by the ResNet18 model. A row orthogonal convolution can be written as a matrix multiplication $\mathbf{y} = \mathcal{K}\mathbf{x}$ of the input $\mathbf{x} \in \mathbb{R}^{CHW}$ with the doubly block-Toeplitz matrix $\mathcal{K} \in \mathbb{R}^{(MH'W') \times (CHW)}$ with the orthogonality condition

$$\langle \mathcal{K}_{i,\cdot}, \mathcal{K}_{j,\cdot} \rangle = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{else} \end{cases} \tag{8}$$

where C and M denote the input and output channels, H and W (H' and W') the spatial dimensions of input $\mathbf{x}$ (output $\mathbf{y}$), and i and j are row indices of $\mathcal{K}$. For our ResNet18 models trained on ImageNet, we compute the pairwise dot product in Eq. 8 between every row of $\mathcal{K}$ in each layer, and present its mean deviation from the orthogonality condition in Fig. 6c. We find that the convolution operators deviate from orthogonality the least when using the Hamming window. We suggest that enforcing orthogonality may be one explanation for the performance increase displayed by windowed convolutions. (See Supplement for further analysis on different models.)

4.5. Adversarial attacks

We test the robustness of baseline and Hamming models (with $M = 6$ layers) trained on the CIFAR-10 dataset against DeepFool [31] (white-box) and spatial transformation [9] (black-box) attacks. DeepFool attacks are iterative attacks designed to minimize the norm of the perturbation while generating examples fast, which we believe is an effective method. Similarly, spatial transformations provide a realistic black-box setting. We generate attacks using the Adversarial Robustness Toolbox [33]. For the DeepFool attacks we use 100 maximum iterations, and for the spatial transformations we use different maximum translation and rotation values as given in Table 2b.

We compare the validation accuracy of baseline and Hamming models with equal number of parameters on the adversarially perturbed CIFAR-10 validation set. For Deep-

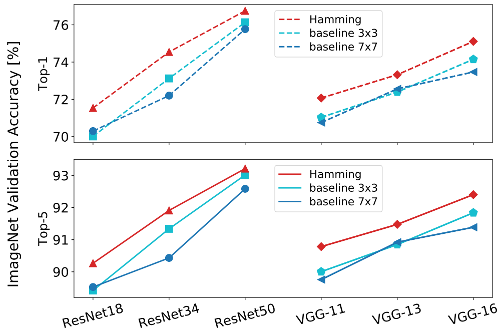


Figure 8. ImageNet validation accuracy for the baseline ResNet and VGG models and their windowed counterparts, where all kernels are replaced with 7 × 7 Hamming windowed kernels.

Model	Top-1 (%)	Top-5 (%)
ResNet18	70.01	89.42
ResNet18 7 × 7	70.30	89.53
ResNet18 + Hamming 7 × 7	**71.54**	**90.27**
ResNet34	73.12	91.34
ResNet34 7 × 7	72.20	90.43
ResNet34 + Hamming 7 × 7	**74.54**	**91.91**
ResNet50	76.14	93.01
ResNet50 7 × 7	75.77	92.58
ResNet50 + Hamming 7 × 7	**76.80**	**93.21**
VGG-11	71.03	90.00
VGG-11 7 × 7	70.75	89.76
VGG-11 + Hamming 7 × 7	**72.07**	**90.78**
VGG-13	72.39	90.85
VGG-13 7 × 7	72.56	90.92
VGG-13 + Hamming 7 × 7	**73.32**	**91.48**
VGG-16	74.15	91.84
VGG-16 7 × 7	73.47	91.39
VGG-16 + Hamming 7 × 7	**75.11**	**92.40**

Table 1. ImageNet validation accuracies in Fig. 8.

DeepFool - Validation Accuracy (%)

Model	Kernel Size		
	5 × 5	7 × 7	9 × 9
baseline	**24.85**±0.34	20.06±0.13	18.24±0.44
Hamming	23.20±0.29	**32.64**±0.39	**27.88**±0.94
baseline-clean	84.93±0.13	83.36±0.16	81.30±0.32
Hamming-clean	**85.77**±0.16	**86.38**±0.12	**86.59**±0.19

(a)

Spatial Transformation - Validation Accuracy (%)

Model	Params		Kernel Size	
	tr	rot	7 × 7	9 × 9
baseline	12.5	22.5	44.59±3.12	38.74±1.37
Hamming			**53.03**±1.91	**52.22**±1.64
baseline	25.0	22.5	31.61±2.87	27.26±1.05
Hamming			**41.44**±2.87	**38.67**±1.46
baseline	25.0	45.0	19.47±0.67	18.13±1.21
Hamming			**26.42**±1.13	**24.55**±1.87

(b)

Table 2. Adversarial robustness in baseline and Hamming models. All results are averaged over 5 runs. (a) Classification accuracy on the CIFAR-10 validation set with and without (clean) perturbations created by the DeepFool attack. (b) Classification accuracy on the CIFAR-10 validation set with spatial transformation attacks for different maximum translation (tr) and rotation (rot) values. Accuracy for unperturbed images is the same as in (a).

Fool attacks, we find that Hamming models with 7 × 7 kernel size provides the best robustness in terms of the decrease in validation accuracy under perturbation (Table 2a). With 5×5 kernels, Hamming models perform worse under DeepFool attacks than baselines, even though the base accuracy on clean samples is higher for Hamming models. For larger kernel sizes, however, the robustness of Hamming models is significantly better. For spatial transform attacks, we find a similar pattern. While validation accuracy decreases across the board for increasing perturbation magnitude, Hamming models with 7×7 and 9×9 kernels are always significantly more robust than the baseline models (Table 2b).

5. Conclusion

We investigate the impact of spectral leakage in the context of CNNs and show that convolutional layers employing small kernel sizes may be susceptible to performance-degrading leakage artifact. As a solution, we propose the use of a standard Hamming window on larger kernels, in line with well-known principles of filter design. We demonstrate enhanced classification accuracy on benchmark datasets, in models with the Hamming window. Finally, we show improved robustness against DeepFool and spatial transformation attacks in windowed CNNs.

This work is based on a simple and well-studied idea, which provides practical benefits in deep networks, highlighting the importance of signal processing fundamentals. We believe our work opens up new research questions regarding other principles of filter design.

Computational cost. Complexity of 2D convolution on a $H{\times}W$ image $\mathcal{O}(HWk^2)$ scales quadratically with kernel size k (or linearly for separable convolutions $\mathcal{O}(2HWk)$ [50]). However, when comparing Hamming vs. 3×3 models, there is a trade-off of increasing kernel size vs. increasing depth to obtain the same accuracy, where the memory load will increase with depth. We show that the use of larger kernels, which are computationally more expensive, but parallelizable compared to deeper networks, is a viable option when the kernels are windowed properly. We note that window functions may provide benefits in domains outside of computer vision, such as audio processing, where larger kernel sizes are common.

Acknowledgements This publication is part of the project "Pixel-free deep learning" (TOP grant with project number 612.001.805) which is financed by the Dutch Research Council (NWO).

References

[1] Zeyuan Allen-Zhu and Yuanzhi Li. Backward feature correction: How deep learning performs deep learning. *arXiv preprint arXiv:2001.04413*, 2020. 1, 3

[2] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402. Springer, 2013. 3

[3] Stan Birchfield. *Image Processing and Analysis*. Cengage Learning, 2016. 2

[4] Joan Bruna and Stéphane Mallat. Invariant scattering convolution networks. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1872–1886, 2013. 2, 3

[5] Minshuo Chen, Yu Bai, Jason D Lee, Tuo Zhao, Huan Wang, Caiming Xiong, and Richard Socher. Towards understanding hierarchical learning: Benefits of neural representations. *Advances in Neural Information Processing Systems, NeurIPS*, 33, 2020. 1, 3

[6] Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. On the relationship between self-attention and convolutional layers. In *International Conference on Learning Representations*, 2019. 1

[7] Arturo Deza, Qianli Liao, Andrzej Banburski, and Tomaso Poggio. Hierarchically local tasks and deep convolutional networks. *arXiv preprint arXiv:2006.13915*, 2020. 1, 2

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 1

[9] Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. Exploring the landscape of spatial robustness. In *International Conference on Machine Learning, ICML*, pages 1802–1811, 2019. 3, 7

[10] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 1625–1634, 2018. 3

[11] R.C. Gonzalez and R.E. Woods. *Digital Image Processing*. Pearson Prentice Hall, 2008. 1, 2

[12] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR*, 2015. 3

[13] Richard W. Hamming. *Digital Filters*. Dover Civil and Mechanical Engineering. Dover Publications, 1998. 1, 2, 4

[14] Stephen Jose Hanson and Lorien Y. Pratt. Comparing biases for minimal network construction with back-propagation. In *Advances in Neural Information Processing Systems 1, [NeurIPS*, pages 177–185, 1988. 2

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 1, 2, 7

[16] 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, NeurIPS*, pages 125–136, 2019. 3

[17] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning, ICML*, volume 37, pages 448–456, 2015. 7

[18] Jorn-Henrik Jacobsen, Jan van Gemert, Zhongyu Lou, and Arnold WM Smeulders. Structured receptive fields in CNNs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2610–2619, 2016. 2, 3

[19] Bernd Jähne. *Digital Image Processing*. Springer, 2005. 1, 2

[20] Judson P Jones and Larry A Palmer. An evaluation of the two-dimensional gabor filter model of simple receptive fields in cat striate cortex. *Journal of neurophysiology*, 58(6):1233–1258, 1987. 1

[21] 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*, pages 14274–14285, 2020. 2

[22] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR*, 2015. 4

[23] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, Department of Computer Science, 04 2009. 6

[24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012. 1, 2

[25] 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. 2, 7

[26] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML*, pages 609–616, 2009. 1, 3

[27] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. In *5th International Conference on Learning Representations, ICLR*, 2017. 3

[28] Marco Loog and François Lauze. Supervised scale-regularized linear convolutionary filters. In *BMVC*, 2017. 2

[29] Shangzhen Luan, Baochang Zhang, Chen Chen, Xianbin Cao, Qixiang Ye, Jungong Han, and Jianzhuang Liu. Gabor Convolutional Networks. *CoRR*, 2017. 3

[30] Michael Mathieu, Mikael Henaff, and Yann LeCun. Fast training of convolutional networks through ffts. In *2nd In-*

ternational Conference on Learning Representations, ICLR 2014, 2014. 2

[31] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition, CVPR*, pages 2574–2582, 2016. 3, 7

[32] Nelson Morgan and Hervé Bourlard. Generalization and parameter estimation in feedforward nets: Some experiments. In *Advances in Neural Information Processing Systems 2, [NeurIPS*, pages 630–637, 1989. 2

[33] Maria-Irina Nicolae, Mathieu Sinn, Minh Ngoc Tran, Beat Buesser, Ambrish Rawat, Martin Wistuba, Valentina Zantedeschi, Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, Ian Molloy, and Ben Edwards. Adversarial robustness toolbox v1.2.0. *CoRR*, 1807.01069, 2018. 7

[34] Bruno A Olshausen and David J Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996. 1, 2

[35] A.V. Oppenheim, R.W. Schafer, J.R. Buck, and L. Lee. *Discrete-time Signal Processing*. Prentice Hall international editions. Prentice Hall, 1999. 1, 2, 3

[36] Edouard Oyallon and Stéphane Mallat. Deep roto-translation scattering for object classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2865–2873, 2015. 2

[37] Edouard Oyallon, Sergey Zagoruyko, Gabriel Huang, Nikos Komodakis, Simon Lacoste-Julien, Matthew B Blaschko, and Eugene Belilovsky. Scattering networks for hybrid representation learning. *IEEE transactions on pattern analysis and machine intelligence*, 2018. 2

[38] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519, 2017. 3

[39] KM Muraleedhara Prabhu. *Window functions and their applications in signal processing*. Taylor & Francis, 2014. 1, 2, 3, 4

[40] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. 7

[41] John L. Semmlow and Benjamin Griffel. *Biosignal and Medical Image Processing, Third Edition*. CRC Press, 2014. 2

[42] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K. Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1528–1540, 2016. 3

[43] Evan Shelhamer, Dequan Wang, and Trevor Darrell. Blurring the line between structure and learning to optimize and adapt receptive fields. *arXiv preprint arXiv:1904.11487*, 2019. 2, 3

[44] Connor Shorten and Taghi M. Khoshgoftaar. A survey on image data augmentation for deep learning. *J. Big Data*, 6:60, 2019. 2

[45] Laurent Sifre and Stéphane Mallat. Rotation, scaling and deformation invariant scattering for texture discrimination. In *CVPR*, 2013. 2

[46] Eero P Simoncelli and Bruno A Olshausen. Natural image statistics and neural representation. *Annual review of neuroscience*, 24(1):1193–1216, 2001. 1, 2

[47] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR*, 2015. 1, 2, 7

[48] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, 2014. 2

[49] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 1–9, 2015. 2

[50] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 2818–2826, 2016. 2, 8

[51] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR*, 2014. 3

[52] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *7th International Conference on Learning Representations, ICLR*, 2019. 3

[53] Jiayun Wang, Yubei Chen, Rudrasis Chakraborty, and Stella X. Yu. Orthogonal convolutional neural networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020*, pages 11502–11512. IEEE, 2020. 5, 7

[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*, pages 5028–5037, 2017. 3

[55] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017. 7

[56] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision, ECCV*, pages 818–833. Springer, 2014. 2

[57] Richard Zhang. Making convolutional networks shift-invariant again. In *Proceedings of the 36th International Conference on Machine Learning, ICML*, volume 97, pages 7324–7334, 2019. 2, 3, 6