From CNNs to Shift-Invariant Twin Wavelet Models

Hubert Leterme[1] Kévin Polisano[2] Valérie Perrier[2] Karteek Alahari[1]

[1]Univ. Grenoble Alpes, CNRS, Inria, Grenoble INP, LJK, 38000 Grenoble, France
[2]Univ. Grenoble Alpes, CNRS, Grenoble INP, LJK, 38000 Grenoble, France

Abstract

We propose a novel antialiasing method to increase shift invariance in convolutional neural networks (CNNs). More precisely, we replace the conventional combination "real-valued convolutions + max pooling" ($\mathbb{R}$Max) by "complex-valued convolutions + modulus" ($\mathbb{C}$Mod), which produce stable feature representations for band-pass filters with well-defined orientations. In our recent work [21], we proved that, for such filters, the two operators yield similar outputs. Therefore, $\mathbb{C}$Mod can be viewed as a stable alternative to $\mathbb{R}$Max. To separate band-pass filters from other freely-trained kernels, in this paper, we designed a "twin" architecture based on the dual-tree complex wavelet packet transform, which generates similar outputs as standard CNNs with fewer trainable parameters. In addition to improving stability to small shifts, our experiments on AlexNet and ResNet showed increased prediction accuracy on natural image datasets such as ImageNet and CIFAR10. Furthermore, our approach outperformed recent antialiasing methods based on low-pass filtering by preserving high-frequency information, while reducing memory usage.

1. Introduction

It has been more than a decade since convolutional neural networks (CNNs) overtook other machine learning methods in large visual recognition tasks, when Krizhevsky *et al*. [20] won the 2012 edition of the ILSVRC challenge on image classification [29]. Since then, some progress has been made on understanding their strengths and limitations. In particular, in order to produce high classification accuracy, we should be able to draw linear boundaries between classes in the feature space. The capability of CNNs to generate such a feature space is therefore of utmost importance. A key property to reach linear separability is the ability to discard or minimize non-discriminative image components. In particular, one could expect feature vectors to be stable with respect to small image deformations such as scaling, rotation or translation. In this paper, we focus on the latter.

A typical CNN architecture contains subsampled convolutions, which are known to be unstable to translations, due to a phenomenon called aliasing [2]. Max pooling, which also comes with subsampling, suffers from the same limitation. Therefore, feature vectors in standard CNNs are not shift invariant, which could penalize the network's accuracy and generalization capability. To overcome this, Zhang [40] proposed an antialiased version of CNNs based on low-pass filtering, called *blur pooling*. This operator is used in two situations. (i) Max pooling operators (MaxPool), which can be decomposed into "Max + Subsampling", are replaced by "Max + BlurPool". (ii) Subsampled convolutions followed by ReLU ("Conv + Subsampling + ReLU") are replaced by "Conv + ReLU + BlurPool". Their approach increased shift invariance and improved accuracy of various networks including AlexNet [20], ResNet [12], DenseNet [15] and MobileNet [14]. Positive results were also obtained on corrupted datasets such as ImageNet-C [13], as well as tiny image datasets such as CIFAR10. However, this is achieved with a significant loss of information.

A question then arises: is it possible to design a non-destructive antialiasing method, and if so, does it further improve accuracy? In a more recent work, Zou *et al*. [42] addressed this question and proposed an adaptive antialiasing approach, called *adaptive blur pooling*, which predicts separate filter weights for each spatial location and output channel. This allows to preserve—to some extent—high-frequency information.

However, the above antialiasing methods come at the cost of increased memory consumption and, for the latter, higher number of trainable parameters. In this paper, we propose an alternative antialiasing method based on complex convolutions, preserving high-frequency information *everywhere*, without increasing the number of parameters or memory consumption. Furthermore, we demonstrate the ability of our approach to improve accuracy.

The proposed method consists in replacing the combinations "real-valued convolution + max pooling" (referred to as $\mathbb{R}$Max) by "complex-valued convolution + modulus" (referred to as $\mathbb{C}$Mod). The complex filters are designed

such that their real and imaginary parts approximately form a 2D Hilbert transform pair [11]. We only do this for high-frequency filters with well-defined orientations, which are naturally present in CNNs when trained on natural image datasets [38]. Unfortunately, the repartition between low- and high-frequency channels is unpredictable, and the former cannot be isolated in a systematic way. To solve this limitation, we consider a mathematical twin mimicking the behavior of standard CNNs with a higher degree of control. In such a model, some freely-trained convolutions are replaced by the real part of a dual-tree complex wavelet packet transform (DT-$\mathbb{C}$WPT) [3], a redundant type of discrete wavelet transform characterized by nearly-analytic filters with various frequencies and orientations. DT-$\mathbb{C}$WPT performs subsampled convolutions, which grants it properties comparable to standard convolution layers with deterministic filters. In this context, replacing $\mathbb{R}$Max by $\mathbb{C}$Mod is straightforward since the complex-valued filters are directly provided by the dual-tree transform.

Our method is motivated by the following theoretical claim. In a recent work [21], we showed that, under specific conditions on the filter's frequency vector and Fourier resolution, $\mathbb{R}$Max and $\mathbb{C}$Mod can produce similar outputs. Furthermore, we proved that the latter operator is stable with respect to small shifts, and deduced a measure of shift invariance for $\mathbb{R}$Max output (*i.e.*, the output of the first max pooling layer in CNNs). This work was essentially theoretical, with limited experiments conducted on a deterministic model solely based on DT-$\mathbb{C}$WPT. However, it lacked applications to tasks such as image classification.

In this paper, we build on this theoretical study, and propose architectures for image classification, considering the $\mathbb{C}$Mod operator as a stable alternative to $\mathbb{R}$Max. We then compare accuracy and shift invariance of $\mathbb{R}$Max- and $\mathbb{C}$Mod-based models. We benchmark our approach against the two antialiasing methods based on blur pooling [40,42]. To do so, we consider models containing "blur-pooled" $\mathbb{R}$Max operators on the one hand, and standalone $\mathbb{C}$Mod operators on the other hand. The code accompanying this work will be made available on GitHub.

2. Related Work

Improving Shift Invariance in CNNs. Following the ideas developed for antialiasing, Chaman and Dokmanic [6] reached perfect shift invariance by using an adaptive, input-dependent subsampling grid, whereas the previous models based on blur pooling rely on fixed grids. This idea was harnessed by Xu *et al*. [37] to get shift equivariance in generative models. However, it is worth noting that, as evidenced by Singla *et al*. [32], shift invariance may alter robustness to other types of adversarial attacks.

Another aspect of shift invariance in CNNs is related to boundary effects. The fact that CNNs can encode the absolute position of an object in the image by exploiting boundary effects was discovered independently by Islam *et al*. [17], and Kayhan and Gemert [18]. This phenomenon is left outside the scope of our paper.

CNNs Meet Wavelet Theory. Several approaches combining CNNs and wavelet transforms have been proposed in the past. Wavelet scattering networks (ScatterNets), by Bruna and Mallat [5], perform cascading wavelet convolutions and nonlinear operations. They produce shift-invariant image representations that are stable to deformation and preserve high-frequency information. Further extensions include roto-translation invariant ScatterNets [28], hybrid ScatterNets combined with fully-trained layers [27] or dictionary learning [39], dual-tree complex wavelet ScatterNets [31], graph ScatterNets [41], learnable ScatterNets via feature map mixing [8] or parametric wavelet filters [10].

These hand-crafted networks are specifically designed to meet some desired properties. As such, they do not intend to reproduce the behavior of standard CNNs. By contrast, in our approach the $\mathbb{C}$Mod operator acts like a proxy for $\mathbb{R}$Max, extracting comparable features with higher stability. Similar to the work discussed earlier in this section, our models are enhanced versions of existing architectures, rather than ad hoc constructions.

Other work improved standard models by augmenting CNNs with wavelet-like filters [7,9,23–25,35,36]. In a similar spirit, Sarwar *et al*. [30] and Ulicny *et al*. [34] built models based on Gabor filters and discrete cosine transforms, respectively. All these approaches aim at either improving the model's predictive power, or reducing its complexity. However, this is achieved with significant modifications to the structure of the networks, and these approaches do not primarily focus on improving stability.

3. Proposed Models

We first explain how the DT-$\mathbb{C}$WPT-based twin models are built. We then describe our antialiasing approach and how it has been compared to related work.

3.1. Notations

We represent CNN feature maps as 2D sequences, denoted by straight capital letters: $\mathrm{X} \in \mathcal{S}$. Indexing is made between square brackets: for any 2D index $\boldsymbol{n} \in \mathbb{Z}^2$, $\mathrm{X}[\boldsymbol{n}] \in \mathbb{R}$ or $\mathbb{C}$. The cross-correlation between X and $\mathrm{V} \in \mathcal{S}$ is defined by $(\mathrm{X} \star \mathrm{V})[\boldsymbol{n}] := \sum_{\boldsymbol{k} \in \mathbb{Z}^2} \mathrm{X}[\boldsymbol{n} + \boldsymbol{k}]\, \mathrm{V}[\boldsymbol{k}]$. The down arrow refers to subsampling: for any $m \in \mathbb{N}^*$, $(\mathrm{X} \downarrow m)[\boldsymbol{n}] := \mathrm{X}[m\boldsymbol{n}]$.

Multidimensional stacks of 2D sequences are represented as bold straight capital letters. A convolution layer with K input channels, L output channels and subsampling $m \in \mathbb{N}^*$ is parameterized by a weight tensor $\mathbf{V} =$

$(\mathrm{V}_{lk})_{lk} \in \mathcal{S}^{L\times K}$ and a bias vector $\boldsymbol{b} = (b_l)_l \in \mathbb{R}^L$. Then, for any multichannel input $\mathbf{X} \in \mathcal{S}^K$, the corresponding output $\mathbf{Y} \in \mathcal{S}^L$ is such that, for any output channel $l \in \{1\,..\,L\}$,

$$\mathrm{Y}_l := \sum_{k=1}^{K} (\mathrm{X}_k \star \mathrm{V}_{lk}) \downarrow m + b_l, \tag{1}$$

where b_l is a bias added to each element of the feature map.

3.2. Wavelet-Based Twin Models (WCNNs)

We describe the general structure of DT-$\mathbb{C}$WPT-based twin CNNs, which we call WCNNs in short. These networks are intended to mimic the behavior of standard architectures after training with natural image datasets.

We consider the first convolution layer of a CNN, as described in (1), after training with ImageNet. For instance, in AlexNet and ResNet architectures, $K = 3$ (RGB input images), $L = 64$, and $m = 4$ and 2, respectively. As widely discussed in the literature [38], a certain number of trained convolution kernels V_{lk} exhibit oscillating patterns with well-defined frequency and orientation. Visual representations of such kernels are provided in Figs. 7a and 9a for AlexNet and ResNet, respectively. We refer to the corresponding output channels as *Gabor channels*. We now build a twin version of this convolution layer as follows.

Let L_{low} and $L_{\mathrm{high}} \in \mathbb{N}$, such that $L_{\mathrm{low}} + L_{\mathrm{high}} = L$. The L_{low} first channels remain, as in the standard model, freely trained, and the corresponding output, denoted by $\mathbf{Y}^{\mathrm{low}} \in \mathcal{S}^{L_{\mathrm{low}}}$, satisfies (1). However, the convolutions producing the remaining L_{high} output feature maps are replaced by a *wavelet block*. The main idea is to constrain the Gabor channels to explicitly compute the real part of DT-$\mathbb{C}$WPT coefficients. Each input image $\mathbf{X}$ goes through the following transformations.

Color Mixing. The RGB components are combined into a single luminance channel. This is implemented by using a 1×1 convolution layer parameterized by a trainable vector $\boldsymbol{\mu} \in \mathbb{R}_+^K$. The output, denoted by $\mathrm{X}^{\mathrm{lum}} \in \mathcal{S}$, satisfies

$$\mathrm{X}^{\mathrm{lum}} := \boldsymbol{\mu}^\top \mathbf{X} = \sum_{k=1}^{K} \mu_k \mathrm{X}_k. \tag{2}$$

Wavelet Packet Decomposition. Then, DT-$\mathbb{C}$WPT with J decomposition stages is performed on $\mathrm{X}^{\mathrm{lum}}$, where $J \in \mathbb{N}^*$ is such that

$$m = 2^{J-1}. \tag{3}$$

To match the subsampling factor m of the standard model, the last decomposition stage is performed without subsampling. We then discard the imaginary part. We refer to this transform as the dual-tree *real* wavelet packet transform (DT-$\mathbb{R}$WPT), which results in $K_{\mathrm{dt}} := 2 \times 4^J$ feature maps:

$$\mathrm{D}_k = \left(\mathrm{X}^{\mathrm{lum}} \star \mathrm{Re}\,\mathrm{W}_k^{(J)}\right) \downarrow 2^{J-1}, \tag{4}$$

where $\mathrm{W}_k^{(J)}$ denotes the k-th dual-tree filter.[1] The power spectrum of this complex filter is concentrated around a specific location in the Fourier domain. Together, the dual-tree filters and their complex conjugates cover the whole frequency plane.

Feature Map Selection. The number of dual-tree feature maps K_{dt} is much greater than the number of Gabor channels L_{high} in general. This stage aims to select filters that contribute the most to the network's predictive power.

First, the low-frequency feature maps D_0 and $\mathrm{D}_{(4^J+1)}$ are discarded. Then, a subset of $K'_{\mathrm{dt}} < K_{\mathrm{dt}}$ feature maps is manually selected and permuted in order to form clusters in the Fourier domain. Considering a (truncated) permutation matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{K'_{\mathrm{dt}} \times K_{\mathrm{dt}}}$, the output of this transformation, denoted by $\mathbf{D}' \in \mathcal{S}^{K'_{\mathrm{dt}}}$, is defined by:

$$\mathbf{D}' := \boldsymbol{\Sigma}\,\mathbf{D}. \tag{5}$$

By design, the feature maps $\mathbf{D}'$ can be sliced into Q groups of channels $\mathbf{D}^{(q)} \in \mathcal{S}^{K_q}$, each of them corresponding to a cluster of band-pass dual-tree filters with neighboring frequencies and orientations. On the other hand, the output of the wavelet block, $\mathbf{Y}^{\mathrm{high}} \in \mathcal{S}^{L_{\mathrm{high}}}$, is also sliced into Q groups of channels $\mathbf{Y}^{(q)} \in \mathcal{S}^{L_q}$. Then, for each group $q \in \{1\,..\,Q\}$, an affine mapping between $\mathbf{D}^{(q)}$ and $\mathbf{Y}^{(q)}$ is performed. It is characterized by a trainable matrix $\boldsymbol{A}^{(q)} \in \mathbb{R}^{L_q \times K_q}$ and bias vector $\boldsymbol{b}^{(q)} \in \mathbb{R}^{L_q}$, such that

$$\mathbf{Y}^{(q)} := \boldsymbol{A}^{(q)}\,\mathbf{D}^{(q)} + \boldsymbol{b}^{(q)}. \tag{6}$$

As in the color mixing stage, this operation is implemented as a 1×1 convolution layer.

Sparse Regularization. For any group $q \in \{1\,..\,Q\}$ and output channel $l \in \{1\,..\,L_q\}$, we want the model to select one and only one wavelet packet feature map within the q-th group. That is, $\mathrm{Y}_l^{(q)} = \alpha_{lk}^{(q)} \mathrm{X}_k^{(q)}$ for some $k \in \{1\,..\,K_q\}$—each row vector $\boldsymbol{\alpha}_l^{(q)}$ of $\boldsymbol{A}^{(q)}$ contains no more than one nonzero element. To enforce this during training, we add a mixed-norm l^1/l^∞-regularizer [22] to the loss function to penalize non-sparse feature map mixing as follows:

$$\mathcal{L} := \mathcal{L}_0 + \sum_{q=1}^{Q} \lambda_q \sum_{l=1}^{L_q} \left(\frac{\left\|\boldsymbol{\alpha}_l^{(q)}\right\|_1}{\left\|\boldsymbol{\alpha}_l^{(q)}\right\|_\infty} - 1 \right), \tag{7}$$

[1] For the sake of computational efficiency, DT-$\mathbb{C}$WPT is performed with successive separable convolutions and linear combinations of wavelet packet feature maps.

where $\mathcal{L}_0$ denotes the standard cross-entropy loss and $\boldsymbol{\lambda} \in \mathbb{R}^Q$ denotes a vector of regularization hyperparameters. Note that the unit bias in (7) serves for interpretability of the regularized loss ($\mathcal{L} = \mathcal{L}_0$ in the desired configuration) but has no impact on training.

Finally, $\mathbf{Y}^{\text{low}}$ and $\mathbf{Y}^{\text{high}}$ are stacked depthwise, which yields the output $\mathbf{Y} \in \mathcal{S}^L$. A schematic representation of the AlexNet-based WCNN architecture (WAlexNet), and the wavelet block upon which it is built, are provided in Figs. 2b and 2d, respectively.

WCNNs *vs*. Standard CNNs. We can show that all output feature maps Y_l satisfy (1). Therefore, WCNNs behave like standard CNNs, with a reduced number of degrees of freedom. For the L_{low} freely-trained output channels, this is straightforward. The remaining L_{high} channels are the outputs of the wavelet block, and are referred to as Gabor channels, by analogy with standard CNNs. For any $l \in \{(L_{\text{low}}+1)\,..\,L\}$, the resulting convolution kernels are as follows:

$$\mathrm{V}_{lk} = \mu_k \operatorname{Re} \widetilde{\mathrm{W}}_l, \tag{8}$$

with $\boldsymbol{\mu}$, the color mixing parameter vector in (2), and

$$\widetilde{\mathrm{W}}_l := \sum_{k=1}^{K'_{\text{dt}}} \alpha_{lk} \mathrm{W}^{(J)}_{\sigma(k)}, \tag{9}$$

with α_{lk} being the coefficients of the block-diagonal matrix $\boldsymbol{A} \in \mathbb{R}^{L_{\text{high}} \times K'_{\text{dt}}}$ generated by $\boldsymbol{A}^{(1)}, \ldots, \boldsymbol{A}^{(Q)}$ as introduced in (6), and $\sigma : K'_{\text{dt}} \to K_{\text{dt}}$ being the permutation function associated with $\boldsymbol{\Sigma}$, introduced in (5).

Figure 1 displays a subset of the kernels $\mathbf{V} \in \mathcal{S}^{64\times 3}$ for the WCNN architecture based on AlexNet, which we refer to as WAlexNet. The kernels are shown as RGB color images, before and after training with ImageNet, for both freely-trained and Gabor channels.

According to (8), given a Gabor channel $l \in \{(L_{\text{low}}+1)\,..\,L\}$, the convolution kernels V_{lk} for $k \in \{1\,..\,K\}$ are equal, up to a multiplicative constant. This property is actually observed in the Gabor channels of standard CNNs: the corresponding RGB kernels roughly appear monochrome (see, for instance, Figs. 7a and 9a). A numerical assessment of this property can by found in [21]. This constraint can be relaxed by performing color mixing *after* DT-$\mathbb{C}$WPT. In this case, the luminance parameter vector $\boldsymbol{\mu}$ differs for each output channel. Numerical experiments on such models are left for future work.

3.3. Complex WCNNs

Our antialiasing method is based on the use of complex convolutions and modulus activations. In WCNNs, the first convolution layer is followed by ReLU and max pooling. If

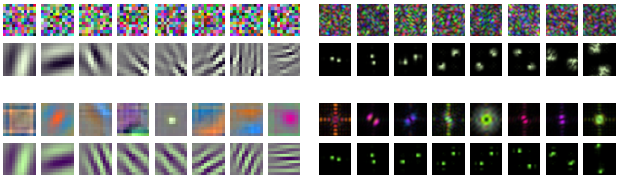


Figure 1. Left: partial representation of WAlexNet's convolution kernels $\mathrm{V} \in \mathcal{S}^{64\times 3}$. A random subset of 8 freely-trained channels and 8 Gabor channels have been selected. The top and bottom images respectively display the kernels before and after training with ImageNet. Right: corresponding power spectra.

we denote by $\mathrm{A}_l \in \mathcal{S}$ the output of the max pooling layer for any given channel $l \in \{1\,..\,L\}$, we get

$$\mathrm{A}_l := \operatorname{MaxPool}\left(\operatorname{ReLU}(\mathrm{Y}_l)\right), \tag{10}$$

with Y_l satisfying (1). Here, for any 2D index $\boldsymbol{n} \in \mathbb{Z}^2$,

$$\operatorname{ReLU}(\mathrm{Y}_l)[\boldsymbol{n}] := \max(0,\, \mathrm{Y}_l[\boldsymbol{n}]); \tag{11}$$

$$\operatorname{MaxPool}(\mathrm{Y}_l)[\boldsymbol{n}] := \max_{\boldsymbol{k}\in\{-1..1\}^2} \mathrm{Y}_l[2\boldsymbol{n}+\boldsymbol{k}]. \tag{12}$$

We now consider the Gabor channels $l \in \{(L_{\text{low}}+1)\,..\,L\}$ from the wavelet block. Max pooling and ReLU can be interchanged. Therefore, according to (3) and (8), expression (10) becomes

$$\mathrm{A}_l = \operatorname{ReLU}(\mathrm{Y}^{\max}_l + b_l), \tag{13}$$

where $\mathrm{Y}^{\max}_l$ is the output of an $\mathbb{R}$Max operator:

$$\mathrm{Y}^{\max}_l := \operatorname{MaxPool}\left\{\left(\mathrm{X}^{\text{lum}} \star \operatorname{Re}\widetilde{\mathrm{W}}_l\right) \downarrow 2^{J-1}\right\}. \tag{14}$$

On the other hand, we consider another operator, $\mathbb{C}$Mod, which is at the heart of our antialiased architecture. Its output, denoted by $\mathrm{Y}^{\text{mod}}_l$, is defined by

$$\mathrm{Y}^{\text{mod}}_l := \left|\left(\mathrm{X}^{\text{lum}} \star \widetilde{\mathrm{W}}_l\right) \downarrow 2^J\right|. \tag{15}$$

Note that $\mathbb{R}$Max and $\mathbb{C}$Mod share the same subsampling factor of 2^J because max pooling is implemented with a subsampling factor of 2.

We assume that, after training, the feature map mixing layer has selected one and only one DT-$\mathbb{C}$WPT channel (see "Sparse Regularization" in Sec. 3.2). In this scenario, (9) becomes

$$\widetilde{\mathrm{W}}_l = \mathrm{W}^{(J)}_k, \tag{16}$$

for some $k \in \{1\,..\,K_{\text{dt}}\}$. Thus, as explained in Sec. 3.2, the filter's power spectrum is concentrated around a specific location in the Fourier domain. Therefore, according to [21], $\mathbb{C}$Mod produces nearly shift-invariant image representations. Moreover, the $\mathbb{R}$Max operator acts as a proxy for $\mathbb{C}$Mod:

$$\mathrm{Y}^{\max}_l \approx \mathrm{Y}^{\text{mod}}_l. \tag{17}$$

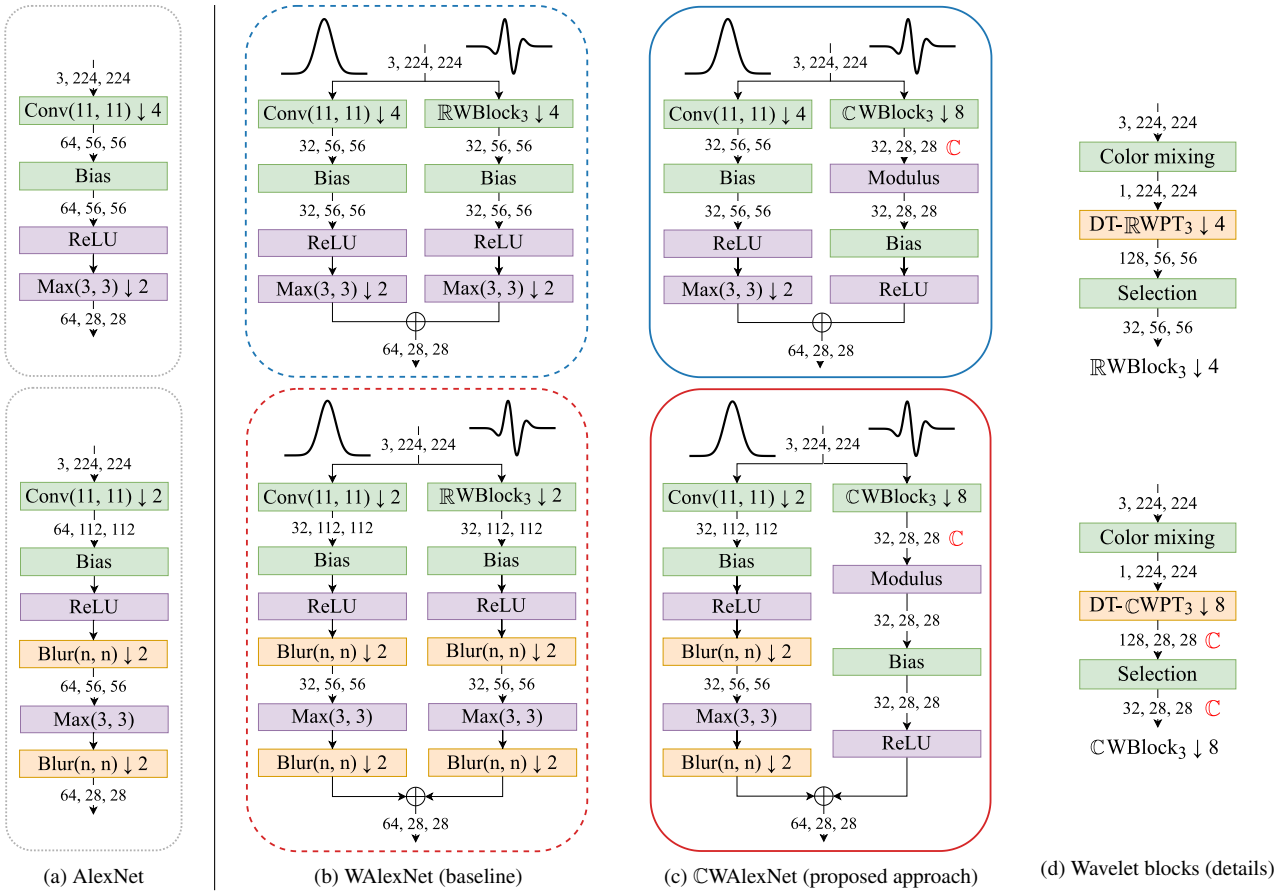


Figure 2. First layers of AlexNet and its variants, corresponding to a convolution layer followed by ReLU and max pooling. The models are framed according to the same colors and line styles as in Figs. 3 and 4. The green modules are the ones containing trainable parameters; the orange and purple modules represent static linear and nonlinear operators, respectively. The bias is represented as a separate module for clarity. The numbers between each module represent the depth (number of channels), height and width of each output. Fig. 2a: freely-trained models. Top: standard AlexNet. Bottom: Zhang's "blurpooled" AlexNet. Fig. 2b: $\mathbb{R}$Max-based twin models (WAlexNet), reproducing the behavior of standard (top) and blurpooled (bottom) AlexNet. The 64 output channels are split into two groups. The left side of each diagram corresponds to the 32 freely-trained, presumably low-frequency output channels, whereas the right side (32 remaining channels) represents $\mathbb{R}\text{Max}^{\text{a}}$ (top) or $\text{Blur}\mathbb{R}\text{Max}^{\text{a}}$ (bottom). Both of them contain a wavelet block, whose structure is detailed in Fig. 2d. Fig. 2c: $\mathbb{C}$Mod-based twin models ($\mathbb{C}$WAlexNet), where $\mathbb{R}\text{Max}^{\text{a}}$ as well as $\text{Blur}\mathbb{R}\text{Max}^{\text{a}}$ are replaced by standalone $\mathbb{C}\text{Mod}^{\text{a}}$. Fig. 2d: details of the wavelet block, in its real and complex configurations (respectively, $\mathbb{R}$WBlock and $\mathbb{C}$WBlock).

The quality of this approximation depends on the filter's frequency and orientation. For some pathological frequencies, the max pooling layer fails to reproduce the behavior of complex modulus. For this reason, $\mathbb{R}$Max may produce unstable representations, as observed in related work.

In order to improve the model's shift invariance, we therefore propose to replace $\mathbb{R}$Max by $\mathbb{C}$Mod for all Gabor output channels. In this family of models, which we call $\mathbb{C}$WCNNs, (13) becomes, for any $l \in \{(L_{\text{low}}+1)\,..\,L\}$,

$$\text{A}'_l = \text{ReLU}(\text{Y}_l^{\text{mod}} + b_l), \tag{18}$$

whereas the first L_{low} channels remain unchanged.

According to (13) and (18), the operators mapping X^{lum} to A_l and A'_l can be interpreted as "activated" versions of $\mathbb{R}$Max and $\mathbb{C}$Mod operators, respectively. As such, they are referred to as $\mathbb{R}\text{Max}^{\text{a}}$ and $\mathbb{C}\text{Mod}^{\text{a}}$.

In practice, the wavelet block is implemented with fully-decimated DT-$\mathbb{C}$WPT (each decomposition stage is performed with a subsampling factor of 2). It is followed by a modulus layer, a bias layer parameterized by a simple vector $\boldsymbol{b}^{\text{high}} \in \mathbb{R}^{L_{\text{high}}}$ and a ReLU activation. All the other layers—including the L_{low} freely-trained channels and the subsequent ReLU and max pooling—continue to follow the principles established in Secs. 3.2 and 3.4. A schematic representation of $\mathbb{C}$WAlexNet is provided in Fig. 2c (top).

	WAlexNet	WResNet
J (decomposition depth)	3	2
K_{dt} (dual-tree filters)	128	32
K'_{dt} (manually selected filters)	94	16
L_{low}, L_{high} (output channels)	32, 32	40, 24

Table 1. Experimental settings for our WCNN twin models. Other details are provided in Appendix A.2.

3.4. WCNNs with Blur Pooling

We benchmark our approach against the antialiasing methods proposed by Zhang [40] and Zou *et al*. [42]. To this end, we consider WCNNs containing blurpooled $\mathbb{R}\mathrm{Max}^{\mathrm{a}}$ on the one hand, and standalone $\mathbb{C}\mathrm{Mod}^{\mathrm{a}}$ on the other hand. Note that $\mathbb{C}\mathrm{Mod}^{\mathrm{a}}$ does not contain blur pooling because it is in itself an antialiased version of $\mathbb{R}\mathrm{Max}^{\mathrm{a}}$. However, for a fair comparison, both $\mathbb{R}$Max- and $\mathbb{C}$Mod-based models use blur pooling in the remaining freely-trained layers.

In what follows, we refer to $\mathrm{Blur}\mathbb{R}\mathrm{Max}^{\mathrm{a}}$ and $\mathrm{ABlur}\mathbb{R}\mathrm{Max}^{\mathrm{a}}$ when talking about antialiased $\mathbb{R}\mathrm{Max}^{\mathrm{a}}$ using, respectively, static and adaptive blur pooling methods. A schematic representation of WAlexNet and $\mathbb{C}$WAlexNet with static blur pooling can be found in Fig. 2b and Fig. 2c, respectively (bottom images).

3.5. Adaptation to ResNet: Batch Normalization

In many recent architectures including ResNet, the bias is computed after an operation called batch normalization (BN) [16]. To build twin WCNNs based on ResNet, which we call WResNet, the bias modules, such as displayed in Fig. 2, are replaced by "BN + bias". However, as shown in Appendix A.1, the modulus layer must be followed by a special type of batch normalization, which we call $\mathbb{C}$ModBN. More specifically, $\mathrm{Y}^{\mathrm{mod}}$ is divided by the square root of its second moment $\mathbb{E}\big[\mathrm{Y}^{\mathrm{mod}^2}\big]$, computed during training over mini-batches. A schematic representation of ResNet-based models, as done in Fig. 2 for AlexNet, is provided in Fig. 6.

4. Experiments

4.1. Experiment Details

ImageNet. We built our WCNN and $\mathbb{C}$WCNN twin models based on AlexNet [20] and ResNet-34 [12] architectures. Their overall design is described in Sec. 3, along with setting details in Tab. 1, whose values were determined empirically from the standard models; further details are provided in Appendix A.2. Regarding benchmarks against antialiasing methods based on blur pooling, Zhang's static approach is tested on both AlexNet and ResNet, whereas Zou *et al*.'s adaptive approach is only tested on ResNet.

To apply blur pooling to $\mathbb{R}\mathrm{Max}^{\mathrm{a}}$, we proceed as follows. Following Zhang's approach, the wavelet block is not antialiased if $m = 2$ as in ResNet, for computational reasons. However, when $m = 4$ as in AlexNet, a blur pooling layer is placed after the ReLU activation. To counterbalance additional subsampling, only the first wavelet decomposition stage (out of $J = 3$) is performed with subsampling, at the cost of increased redundancy. This kind of wavelet transform is qualified as (partially) stationary, as introduced in Nason and Silverman [26]. In any case, as in the standard models, MaxPool is replaced by "Max + BlurPool".

When antialiasing our models we used filters of size 4 and 3 for static and adaptive blur pooling methods respectively, which are the default values in publicly available repositories [40, 42]. Besides, DT-$\mathbb{C}$WPT decompositions are performed with Q-shift orthogonal filters of length 10 as introduced by Kingsbury [19].

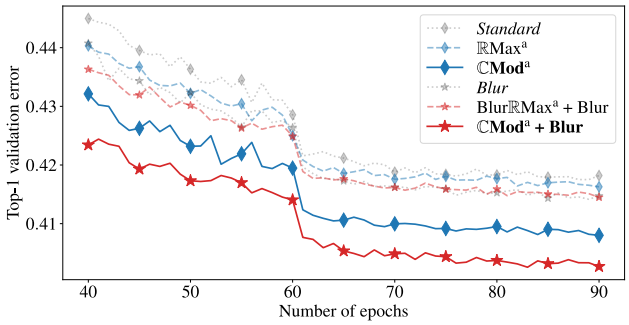
Our models were trained on the ImageNet ILSVRC2012 dataset [29], following the standard procedure provided by PyTorch [1]. Moreover, we set aside 100K images from the training set—100 per class—in order to compute the top-1 error rate after each training epoch.

CIFAR-10. We also trained ResNet-18, ResNet-34 and their variants on the CIFAR-10 dataset. Training was performed on 300 epochs, with an initial learning rate equal to 0.1, decreased by a factor of 10 every 100 epochs. As for ImageNet, we set aside 5 000 images out of 50K to compute accuracy during the training phase. Given the images of small size in this dataset (32×32 pixels), feature extraction can be performed efficiently with a reduced number of layers. For this reason, the first layers ("convolution + max pooling") arguably have a higher influence on the overall predictive power. We therefore expect to clearly highlight the benefits of our approach on this specific task.

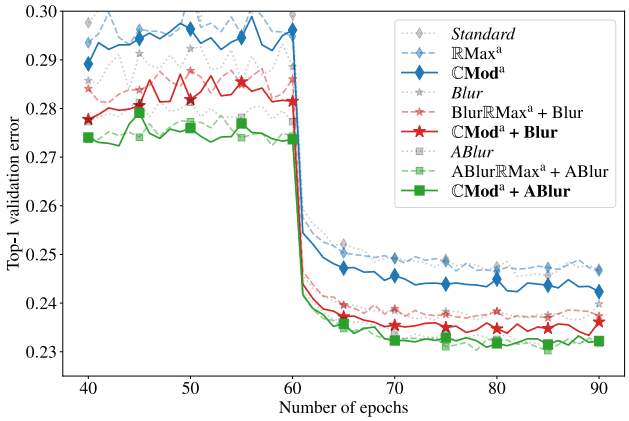
4.2. Evaluation Metrics

Classification Accuracy. Classification accuracy was computed on the standard ImageNet evaluation set (50K images). We followed the *ten-crops* procedure [20]: predictions are made over 10 patches extracted from each input image, and the softmax outputs are averaged to get the overall prediction. We also considered center crops of size 224 for *one-crop* evaluation. In both cases, we used top-1-5 error rates. For CIFAR-10 (10K images) evaluation, we measured the top-1 error rate with one- and ten-crops.

Measuring Shift Invariance. For each image in the ImageNet evaluation set, we extracted several patches of size 224, each of which being shifted by 0.5 pixel along a given axis. We then compared their outputs in order to measure the model's robustness to shifts. This was done by computing the Kullback-Leibler (KL) divergence between output vectors—which, under certain hypotheses, can be inter-



(a) AlexNet-based models



(b) ResNet-based models

Figure 3. Evolution of the top-1 validation error along training with ImageNet, for AlexNet and ResNet. These plots display the twin WCNNs without blur pooling (blue diamonds), with static (red stars) and adaptive (green squares) blur pooling. The ℂMod-based models appear in solid lines. Besides, standard AlexNet and ResNet, upon which the twin models are built, appear in gray.

preted as probability distributions [4, pp. 205-206]. This metrics is intended for visual representation.

In addition, we measured the mean flip rate (mFR) between predictions, as introduced by Hendrycks and Dietterich [13]. For each direction (vertical, horizontal and diagonal), we measured the mean frequency upon which two shifted input images yield different top-1 predictions, for shift distances varying from 1 to 8 pixels. We then normalized the results with respect to AlexNet's mFR, and averaged over the three directions.

We repeated the procedure for the models trained on CIFAR-10. This time, we extracted patches of size 32×32 from the evaluation set, and computed mFR for shifts varying from 1 to 4 pixels. Besides, normalization was performed with respect to ResNet-18's mFR.

4.3. Results and Discussion

Validation Curves. Top-1 accuracy measured along training with ImageNet is presented in Fig. 3. Regarding AlexNet (Fig. 3a), we notice significant improvements when ℂMod^a replaces ℝMax^a (blue diamonds), or BlurℝMax^a (red stars). In fact, our ℂMod-based approach alone (blue diamonds, solid line) suffices to perform better

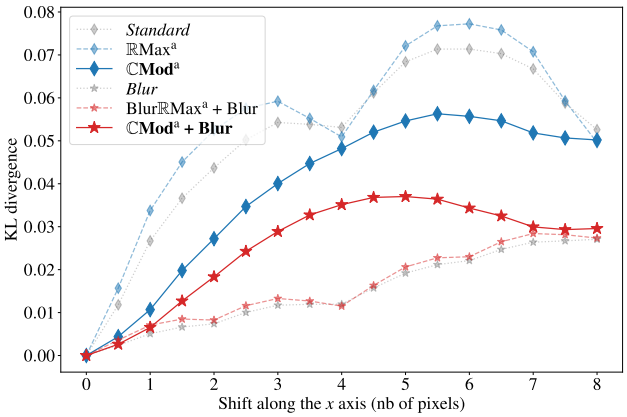


Figure 4. AlexNet-based models: mean KL divergence between the outputs of a reference image versus shifted images.

Model	One-crop top-1	One-crop top-5	Ten-crops top-1	Ten-crops top-5	Shifts mFR
	AlexNet				
Standard	*45.3*	*22.2*	*41.3*	*19.3*	*100.0*
ℝMax^a	44.9	21.8	40.8	19.0	101.4
ℂMod^a	**44.3**	**21.3**	**40.2**	**18.5**	**88.0**
Blur	*44.8*	*22.0*	*41.1*	*19.1*	*58.1*
BlurℝMax^a + Blur	44.6	21.9	40.6	19.0	**59.2**
ℂMod^a + Blur	**43.6**	**20.9**	**39.5**	**17.9**	71.0
	ResNet-34				
Standard	*27.6*	*9.2*	*24.8*	*7.7*	*78.1*
ℝMax^a	27.4	9.2	24.7	7.6	77.2
ℂMod^a	**27.2**	**9.0**	**24.4**	**7.4**	**73.1**
Blur	*26.5*	*8.7*	*24.1*	*7.3*	*60.3*
BlurℝMax^a + Blur	**26.6**	8.7	24.3	**7.3**	62.7
ℂMod^a + Blur	**26.6**	**8.6**	**24.0**	**7.3**	**61.5**
ABlur	*26.1*	*8.3*	*23.5*	*7.0*	*60.8*
ABlurℝMax^a + ABlur	**26.0**	**8.2**	**23.6**	**6.9**	**62.1**
ℂMod^a + ABlur	26.1	**8.2**	23.7	7.0	63.1

Table 2. Evaluation metrics on ImageNet (%).

Model	ResNet-18 1crp	ResNet-18 10crp	ResNet-18 shft	ResNet-34 1crp	ResNet-34 10crps	ResNet-34 shft
Standard	*14.9*	*10.8*	*100.0*	*15.2*	*10.9*	*100.3*
ℝMax^a	14.2	10.3	92.4	14.5	10.5	99.2
ℂMod^a	**13.8**	**9.6**	**88.8**	**12.9**	**9.2**	**93.0**
Blur	*14.2*	*10.4*	*87.7*	*15.7*	*11.6*	*88.2*
BlurℝMax^a + Blur	13.1	9.7	**84.6**	13.2	9.9	85.6
ℂMod^a + Blur	**12.3**	**8.9**	85.7	**12.4**	**9.1**	**83.7**
ABlur	*14.6*	*11.0*	*90.9*	*16.3*	*12.8*	*91.9*
ABlurℝMax^a + ABlur	14.5	11.0	86.5	14.0	10.4	93.3
ℂMod^a + ABlur	**12.8**	**9.7**	**81.7**	**12.8**	**9.2**	**86.6**

Table 3. Evaluation metrics on CIFAR-10 (%): top-1 error rate using one- and ten-crops methods ("1crp" and "10crp"); and mFR measuring shift invariance ("shft").

than both WAlexNet and blurpooled WAlexNet (blue diamonds and red stars, dashed lines).

Now, when looking at ResNet (Fig. 3b), similar improvements are observed, however to a lesser extent. More precisely, top-1 validation error decreases when $\mathbb{C}\text{Mod}^{\text{a}}$ replaces $\mathbb{R}\text{Max}^{\text{a}}$ (blue diamonds), or $\text{Blur}\mathbb{R}\text{Max}^{\text{a}}$ (red stars). However, no visible improvement seems to occur when $\text{ABlur}\mathbb{R}\text{Max}^{\text{a}}$ is replaced by $\mathbb{C}\text{Mod}^{\text{a}}$ (green squares).

Finally, we notice that the training curves for WCNNs (colored dashed lines) closely follow those of standard CNNs (gray dotted lines). This is an expected behavior since the former models are designed to mimic the behavior of the latter. However, the training curve for WAlexNet (blue diamonds, Fig. 3a) is below standard AlexNet for a large part of training, before converging toward the end. This can be explained by the more constrained architecture, which can lead to faster training.

Accuracy Scores on Evaluation Sets. Error rates computed on the evaluation sets are provided in Tab. 2 for ImageNet and Tab. 3 for CIFAR-10.

The figures reported in Tab. 2 are consistent with our observations from Fig. 3. We observe a significant increase in accuracy for AlexNet, and mixed results for ResNet. More precisely, replacing $\mathbb{R}\text{Max}^{\text{a}}$ (without blur pooling) by $\mathbb{C}\text{Mod}^{\text{a}}$ clearly improves accuracy. The gain is less pronounced, albeit still positive, when replacing $\text{Blur}\mathbb{R}\text{Max}^{\text{a}}$ (Zhang's method) by $\mathbb{C}\text{Mod}^{\text{a}}$. However, $\mathbb{C}\text{Mod}^{\text{a}}$ slightly degrades performances over $\text{ABlur}\mathbb{R}\text{Max}^{\text{a}}$ (Zou *et al.*'s method). Arguably, the higher gains obtained on AlexNet, compared to ResNet, are linked to the higher impact of early layers on the network's accuracy, due to the higher subsampling factor. Moreover, ResNet models are deeper than AlexNet, limiting the relative influence of our antialiasing method.

Nevertheless, our method nearly achieves, or even exceeds, the predictive power of WCNNs antialiased with blur pooling methods, with a significantly reduced memory footprint—more details on this is provided in Appendix A.4. Moreover, in the adaptive blur pooling scenario, additional trainable parameters, and therefore computational complexity, are required (an adaptive blur pooling layer typically contains more than 40K parameters). Conversely, in our method, $\mathbb{C}\text{Mod}^{\text{a}}$ does not contain any additional parameter when compared with $\mathbb{R}\text{Max}^{\text{a}}$.

Finally, when trained on CIFAR-10, all models achieve significant gains in accuracy when replacing $\mathbb{R}\text{Max}^{\text{a}}$, $\text{Blur}\mathbb{R}\text{Max}^{\text{a}}$, or even $\text{ABlur}\mathbb{R}\text{Max}^{\text{a}}$, by $\mathbb{C}\text{Mod}^{\text{a}}$. As for AlexNet with ImageNet, this is explained by a higher impact of early layers on the network's accuracy. In the CIFAR case though, this is due to the small size of input images.

Shift Invariance (KL Divergence). The mean KL divergence between outputs of shifted images are plotted in Fig. 4 for AlexNet trained on ImageNet.

In standard and twin WAlexNet (gray and blue diamonds, dashed lines), when the input image is shifted by 4 pixels, the output of the first convolution layer is strictly shifted by one pixel. The first layer is therefore equivariant to a 4-pixel shift. Consequently, any divergence between outputs is due to the action of deeper layers—and probably to boundary effects. Likewise, when the shift is equal to 8 pixels, equivariance applies to the output of the max pooling layer. When replacing $\mathbb{R}\text{Max}^{\text{a}}$ by $\mathbb{C}\text{Mod}^{\text{a}}$ (blue diamonds, solid line), a similar 8-pixel-equivariance applies to the output of the modulus layer. However, what happens in between depends on the chosen model. We observe that $\mathbb{C}$WAlexNet smoothens the curve, avoiding the "bumps" observed in non-antialiased models.

On the other hand, standard and twin WAlexNet antialiased with blur pooling (gray and red stars, dashed lines) have their curves considerably flattened compared to non-antialiased models, or even to $\mathbb{C}$WAlexNet without blur pooling (gray and blue diamonds). This demonstrates the efficiency of Zhang's blur pooling method. Contrary to the previous models, replacing $\text{Blur}\mathbb{R}\text{Max}^{\text{a}}$ by $\mathbb{C}\text{Mod}^{\text{a}}$ (red stars, solid line) degrades shift invariance, as witnessed by the bell-shaped curve. And yet, the corresponding classifier is significantly more accurate. This can be explained as follows. Blur pooling methods are fundamentally based on low-pass filtering, causing significant loss of information. By contrast, our antialiasing method is designed to keep all high-frequency information (up to a phase shift), which may contain discriminant features. Therefore, a tradeoff between information preserving and perfect shift invariance seems necessary to achieve the best performances. Note that KL divergence at 8-pixel shifts is lower for blurpooled models. This is because deeper layers are also antialiased, transforming shift equivariance into near shift invariance.

Shift Invariance (Mean Flip Rate). The mean flip rate for shifted inputs is reported in Tab. 2 for ImageNet (AlexNet and ResNet-34) and Tab. 3 for CIFAR-10 (ResNet-18 and 34). In most situations, replacing standalone or blurpooled $\mathbb{R}\text{Max}^{\text{a}}$ by $\mathbb{C}\text{Mod}^{\text{a}}$ leads to decreased mFR, and therefore improved shift invariance. The two exceptions are blurpooled AlexNet, for which an explanation is provided above, and adaptive-blurpooled ResNet on ImageNet.

5. Conclusion and Perspectives

Shift invariance can lead to increased accuracy—this is what is suggested by this work and several others introducing antialiasing techniques in CNNs. The method proposed in this paper, which consists in replacing $\mathbb{R}\text{Max}$ operators

by $\mathbb{C}$Mod, preserves high-frequency information. By doing so, we reached or even outperformed previous methods based on low-pass filtering, while limiting memory consumption and, to some extent, computational complexity. An interesting research direction would be to extend this method to deeper layers, or adapt its principles to other deep learning architectures.

Acknowledgements

This work has been partially supported by the LabEx PERSYVAL-Lab (ANR-11-LABX-0025-01) funded by the French program Investissement d'avenir, as well as the ANR grants MIAI (ANR-19-P3IA-0003) and AVENUE (ANR-18-CE23-0011). Most of the computations presented in this paper were performed using the GRICAD infrastructure,[2] which is supported by Grenoble research communities.

References

[1] PyTorch "examples" repository available at https://github.com/pytorch/examples/tree/main/imagenet. 6

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

[3] Ilker Bayram and Ivan W. Selesnick. On the Dual-Tree Complex Wavelet Packet and M-Band Transforms. *IEEE Transactions on Signal Processing*, 56(6):2298–2310, June 2008. 2, 11

[4] Christopher M. Bishop and Tom M. Mitchell. *Pattern Recognition and Machine Learning*. Springer, 2014. 7

[5] Joan Bruna and Stéphane Mallat. Invariant Scattering Convolution Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1872–1886, May 2013. 2

[6] Anadi Chaman and Ivan Dokmanic. Truly Shift-Invariant Convolutional Neural Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3773–3783, 2021. 2

[7] Fergal Cotter and Nick Kingsbury. Deep Learning in the Wavelet Domain. November 2018. 2

[8] Fergal Cotter and Nick Kingsbury. A Learnable Scatternet: Locally Invariant Convolutional Layers. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 350–354, September 2019. 2

[9] Shin Fujieda, Kohei Takayama, and Toshiya Hachisuka. Wavelet Convolutional Neural Networks for Texture Classification. *arXiv:1707.07394*, July 2017. 2

[10] Shanel Gauthier, Benjamin Thérien, Laurent Alsène-Racicot, Muawiz Chaudhary, Irina Rish, Eugene Belilovsky, Michael Eickenberg, and Guy Wolf. Parametric Scattering Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5749–5758, 2022. 2

[11] J.P. Havlicek, J.W. Havlicek, and A.C. Bovik. The analytic image. In *Proceedings of International Conference on Image Processing*, volume 2, pages 446–449 vol.2, October 1997. 2

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016. 1, 6

[13] Dan Hendrycks and Thomas Dietterich. Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. In *International Conference on Learning Representations*, March 2019. 1, 7

[14] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv:1704.04861*, April 2017. 1

[15] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2261–2269, 2017. 1

[16] 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*, pages 448–456. PMLR, June 2015. 6

[17] Md Amirul Islam, Sen Jia, and Neil D. B. Bruce. How Much Position Information Do Convolutional Neural Networks Encode? In *International Conference on Learning Representations*, January 2020. 2

[18] 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

[19] Nick Kingsbury. Design of Q-shift complex wavelets for image processing using frequency domain energy minimization. In *Proceedings International Conference on Image Processing*, volume 1, pages I–1013, 2003. 6

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, May 2017. 1, 6

[21] Hubert Leterme, Kévin Polisano, Valérie Perrier, and Karteek Alahari. On the Shift Invariance of Max Pooling Feature Maps in Convolutional Neural Networks. *arXiv:2209.11740*, September 2022. 1, 2, 4, 11

[22] Jun Liu and Jieping Ye. Efficient L1/Lq Norm Regularization. *arXiv:1009.4766*, September 2010. 3

[2] https://gricad.univ-grenoble-alpes.fr

[23] Lin Liu, Jianzhuang Liu, Shanxin Yuan, Gregory Slabaugh, Aleš Leonardis, Wengang Zhou, and Qi Tian. Wavelet-Based Dual-Branch Network for Image Demoiréing. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 86–102, 2020. 2

[24] Hongya Lu, Haifeng Wang, Qianqian Zhang, Daehan Won, and Sang Won Yoon. A Dual-Tree Complex Wavelet Transform Based Convolutional Neural Network for Human Thyroid Medical Image Segmentation. In *IEEE International Conference on Healthcare Informatics (ICHI)*, pages 191–198, 2018. 2

[25] Shangzhen Luan, Chen Chen, Baochang Zhang, Jungong Han, and Jianzhuang Liu. Gabor Convolutional Networks. *IEEE Transactions on Image Processing*, 27(9):4357–4366, May 2018. 2

[26] G. P. Nason and B. W. Silverman. The Stationary Wavelet Transform and some Statistical Applications. In Anestis Antoniadis and Georges Oppenheim, editors, *Wavelets and Statistics*, Lecture Notes in Statistics, pages 281–299. Springer, 1995. 6

[27] Edouard Oyallon, Eugene Belilovsky, and Sergey Zagoruyko. Scaling the Scattering Transform: Deep Hybrid Networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5618–5627, 2017. 2

[28] Edouard Oyallon and Stephane 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

[29] 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*, 115(3):211–252, April 2015. 1, 6

[30] Syed Shakib Sarwar, Priyadarshini Panda, and Kaushik Roy. Gabor filter assisted energy efficient fast learning Convolutional Neural Networks. In *2017 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, pages 1–6, 2017. 2

[31] Amarjot Singh and Nick Kingsbury. Dual-Tree wavelet scattering network with parametric log transformation for object classification. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2017. 2

[32] Vasu Singla, Songwei Ge, Basri Ronen, and David Jacobs. Shift Invariance Can Reduce Adversarial Robustness. In *Advances in Neural Information Processing Systems*, volume 34, pages 1858–1871, 2021. 2

[33] Antonio Torralba and Aude Oliva. Statistics of natural image categories. *Network: Computation in Neural Systems*, 14(3):391–412, January 2003. 11

[34] Matej Ulicny, Vladimir A. Krylov, and Rozenn Dahyot. Harmonic Networks for Image Classification. In *British Machine Vision Conference*, page 202, 2019. 2

[35] Travis Williams and Robert Li. Advanced Image Classification Using Wavelets and Convolutional Neural Networks. In *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 233–239, 2016. 2

[36] Travis Williams and Robert Li. Wavelet Pooling for Convolutional Neural Networks. In *International Conference on Learning Representations*, 2018. 2

[37] Jin Xu, Hyunjik Kim, Thomas Rainforth, and Yee Teh. Group Equivariant Subsampling. In *Advances in Neural Information Processing Systems*, volume 34, pages 5934–5946, 2021. 2

[38] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*, pages 3320–3328, 2014. 2, 3

[39] John Zarka, Louis Thiry, Tomás Angles, and Stéphane Mallat. Deep Network Classification by Scattering and Homotopy Dictionary Learning. In *International Conference on Learning Representations*, 2020. 2

[40] Richard Zhang. Making Convolutional Networks Shift-Invariant Again. In *International Conference on Machine Learning*, pages 7324–7334. PMLR, May 2019. 1, 2, 6, 14

[41] Dongmian Zou and Gilad Lerman. Graph convolutional neural networks via scattering. *Applied and Computational Harmonic Analysis*, 49(3):1046–1074, November 2020. 2

[42] Xueyan Zou, Fanyi Xiao, Zhiding Yu, and Yong Jae Lee. Delving Deeper into Anti-aliasing in ConvNets. In *British Machine Vision Conference*, August 2020. 1, 2, 6, 14

A. Appendix

A.1. Batch Normalization in ResNet

This section complements Sec. 3.5. In many recent architectures including ResNet, the bias (see Fig. 2) is replaced by an affine batch normalization layer (BN). In this section, we show how to adapt our approach to this context.

A BN layer is parameterized by trainable weight and bias vectors, respectively denoted by $\boldsymbol{a}$ and $\boldsymbol{b} \in \mathbb{R}^L$. In the remaining of the section, we consider $\mathrm{X}^{\mathrm{lum}}$, introduced in (2), as a discrete stochastic process. Then, (10) is replaced by

$$\mathrm{A}_l := \mathrm{MaxPool}\left\{ \mathrm{ReLU}\left(a_l \cdot \frac{\mathrm{Y}_l - \mathbb{E}_m[\mathrm{Y}_l]}{\sqrt{\mathbb{V}_m[\mathrm{Y}_l] + \varepsilon}} + b_l \right) \right\}, \quad (19)$$

with Y_l satisfying (1). In the above expression, we have introduced $\mathbb{E}_m(\mathrm{Y}_l) \in \mathbb{R}$ and $\mathbb{V}_m(\mathrm{Y}_l) \in \mathbb{R}_+$, which respectively denote the mean expected value and variance of $\mathrm{Y}_l[\boldsymbol{n}]$, for indices $\boldsymbol{n}$ contained in the support of Y_l, denoted by $\mathrm{supp}(\mathrm{Y}_l)$. Let us denote by $N \in \mathbb{N}^*$ the support size of input images. Therefore, if the filter's support size N_{filt} is much smaller that N, then $\mathrm{supp}(\mathrm{Y}_l)$ is roughly of size

N/m. We then define the above quantities as follows:

$$\mathbb{E}_m[\mathrm{Y}_l] := \frac{m^2}{N^2} \sum_{\boldsymbol{n} \in \mathbb{Z}^2} \mathbb{E}[\mathrm{Y}_l[\boldsymbol{n}]]; \tag{20}$$

$$\mathbb{V}_m[\mathrm{Y}_l] := \frac{m^2}{N^2} \sum_{\boldsymbol{n} \in \mathbb{Z}^2} \mathbb{V}[\mathrm{Y}_l[\boldsymbol{n}]]. \tag{21}$$

In practice, estimators are computed over a minibatch of images, hence the layer's denomination. Besides, $\varepsilon > 0$ is a small constant added to the denominator for numerical stability. For the sake of concision, we now assume that $\boldsymbol{a} = \mathbf{1}$. Extensions to other multiplicative factors is straightforward.

Proposition 1. *We assume that the Fourier transform of $\widetilde{\mathrm{W}}_l$, such as introduced in (9), is supported in a region of size $\kappa \times \kappa$ which does not contain the origin (band-pass filter). If, moreover, $\kappa \leq \frac{2\pi}{m}$, then*

$$\sum_{\boldsymbol{n} \in \mathbb{Z}^2} \mathrm{Y}_l[\boldsymbol{n}] = 0. \tag{22}$$

Proof. This proposition takes advantage of Shannon's sampling theorem. A similar reasoning can be found in the proof of Theorem 1 in [21]. □

As before, we assume that (16) is satisfied for each Gabor channel $l \in \{(L_{\mathrm{low}} + 1) \,..\, L\}$ (*i.e.*, only one DT-ℂWPT feature map has been selected). The power spectrum of DT-ℂWPT filters cannot be exactly zero on regions with nonzero measure, since they are finitely supported. However, we can reasonably assume that it is concentrated within a region of size $\pi/2^{J-1} = \pi/m$, as explained in [21]. Therefore, since we have discarded low-pass filters, the conditions of Prop. 1 are approximately met for $\widetilde{\mathrm{W}}_l$.

We now assume that (22) is satisfied. Moreover, we assume that $\mathbb{E}[\mathrm{Y}_l[\boldsymbol{n}]]$ is constant for any $\boldsymbol{n} \in \mathrm{supp}(\mathrm{Y}_l)$. Aside from boundary effects, this is true if $\mathbb{E}[\mathrm{X}^{\mathrm{lum}}[\boldsymbol{n}]]$ is constant for any $\boldsymbol{n} \in \mathrm{supp}(\mathrm{X}^{\mathrm{lum}})$.[3] We then get, for any $\boldsymbol{n} \in \mathbb{Z}^2$, $\mathbb{E}[\mathrm{Y}_l[\boldsymbol{n}]] = 0$. Therefore, interchanging max pooling and ReLU yields the normalized version of (13):

$$\mathrm{A}_l = \mathrm{ReLU}\left(\frac{\mathrm{Y}_l^{\max}}{\sqrt{\mathbb{E}_m[\mathrm{Y}_l^2] + \varepsilon}} + b_l\right). \tag{23}$$

As in Sec. 3.3, we replace $\mathrm{Y}_l^{\max}$ by $\mathrm{Y}_l^{\mathrm{mod}}$ for any Gabor channel $l \in \{(L_{\mathrm{low}} + 1) \,..\, L\}$, which yields the normalized version of (18):

$$\mathrm{A}'_l := \mathrm{ReLU}\left(\frac{\mathrm{Y}_l^{\mathrm{mod}}}{\sqrt{\mathbb{E}_m[\mathrm{Y}_l^2] + \varepsilon}} + b_l\right). \tag{24}$$

[3] This property is a rough approximation for natural images. In practice, the main subject is generally located at the center, the sky at the top, *etc*. These are sources of variability for color and luminance distributions across images, as discussed by Torralba and Oliva [33].

Implementing (24) as a deep learning architecture is cumbersome because Y_l needs to be explicitly computed and kept in memory, in addition to $\mathrm{Y}_l^{\mathrm{mod}}$. Instead, we want to express the second-order moment $\mathbb{E}_m[\mathrm{Y}_l^2]$ as a function of $\mathrm{Y}_l^{\mathrm{mod}}$. To this end, we state the following proposition.

Proposition 2. *If we restrict the conditions of Prop. 1 to $\kappa \leq \pi/m$, we have*

$$\|\mathrm{Y}_l\|_2^2 = 2\left\|\mathrm{Y}_l^{\mathrm{mod}}\right\|_2^2. \tag{25}$$

Proof. This result, once again, takes advantage of the Shannon's sampling theorem. The proof of our Proposition 3 in [21] is based on similar arguments. □

As for Prop. 1, the conditions of Prop. 2 are approximately met. We therefore assume that (25) is satisfied, and (24) becomes

$$\mathrm{A}'_l := \mathrm{ReLU}\left(\frac{\mathrm{Y}_l^{\mathrm{mod}}}{\sqrt{\frac{1}{2}\mathbb{E}_{2m}[{\mathrm{Y}_l^{\mathrm{mod}}}^2] + \varepsilon}} + b_l\right). \tag{26}$$

Therefore, the bias layer following the modulus operator in Fig. 2c is replaced by a modified batch normalization layer implementing (26), which we call ℂModBN. The second-order moment of ${\mathrm{Y}_l^{\mathrm{mod}}}^2$ is computed on feature maps which are twice smaller than Y_l in both directions—hence the index "$2m$" in (26), which is the subsampling factor for the ℂMod operator.

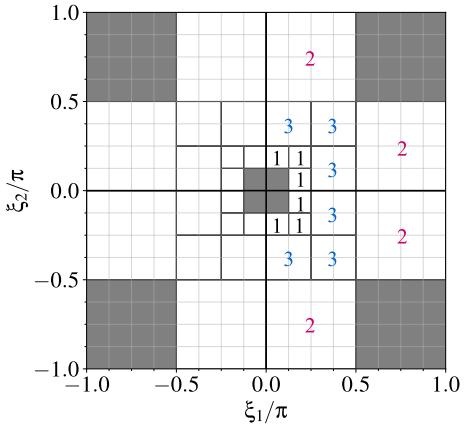
Schematic representations of ℝMax- and ℂMod-based ℂWResNet are provided in Fig. 6.

A.2. Setting Details for WCNNs

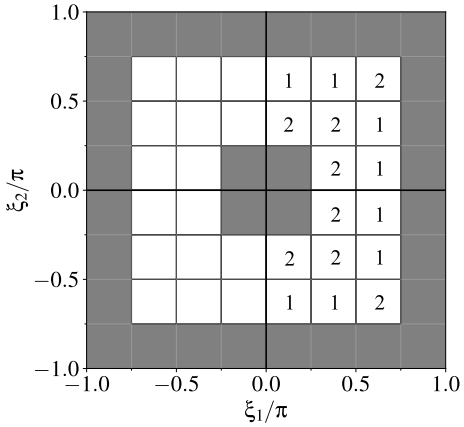
In this section, we provide further information that complements the experimental details presented in Sec. 4.1 and Tab. 1.

According to (3), the decomposition depth J is determined by the subsampling factor m, which is equal to 4 in AlexNet and 2 in ResNet. We then get the number of dual-tree filters $K_{\mathrm{dt}} := 2 \times 4^J$.

We then manually selected $K'_{\mathrm{dt}} < K_{\mathrm{dt}}$ filters. In particular, we removed the two low-pass filters, which are outside the scope of our theoretical study. Besides, for computational reasons, in WAlexNet we removed 32 "extremely" high-frequency filters which are clearly absent from the standard model (see Fig. 5a). Finally, in WResNet we removed the 14 filters whose characteristic frequencies lie close to an edge of the Fourier domain $[-\pi, \pi]^2$ (see Fig. 5b). These filters indeed have a poorly-defined orientation, since a small fraction of their energy is located at the far end of the Fourier domain [3, see Fig. 1, "Proposed DT-ℂWPT"]. Therefore, they somewhat exhibit a checkerboard pattern.



(a) WAlexNet ($J = 3$)



(b) WResNet ($J = 2$)

Figure 5. Mapping scheme from DT-$\mathbb{C}$WPT feature maps $\mathbf{D} \in \mathcal{S}^{K_{\mathrm{dt}}}$ to the wavelet block's output $\mathbf{Y}^{\mathrm{high}} \in \mathcal{S}^{L_{\mathrm{high}}}$. Each wavelet feature map is symbolized by a small faint square in the Fourier domain, where its energy is mainly located. The gray areas show the feature maps which have been manually discarded. Elsewhere, each group of feature maps $\mathbf{D}^{(q)} \in \mathcal{S}^{K_q}$ is symbolized by a dark frame (in ResNet, K_q is always equal to 1). For each group $q \in \{1\,..\,Q\}$, a number indicates how many output channels L_q are assigned to it. The colored numbers in Fig. 5a refer to groups on which we have applied l^∞/l^1-regularization. Note that, without any loss of information, only half of the filters are considered when inputs are real-valued (in our example, positive x-values, but we could also have considered the complex conjugates).

As explained in Sec. 3.2, once the DT-$\mathbb{C}$WPT feature maps have been manually selected, the output $\mathbf{D}' \in \mathcal{S}^{K'_{\mathrm{dt}}}$ is sliced into Q groups of channels $\mathbf{D}^{(q)} \in \mathcal{S}^{K_q}$. For each group q, a depthwise linear mapping from $\mathbf{D}^{(q)}$ to a bunch of output channels $\mathbf{Y}^{(q)} \in \mathcal{S}^{L_q}$ is performed. Finally, the wavelet block's output feature maps $\mathbf{Y}^{\mathrm{high}} \in \mathcal{S}^{L_{\mathrm{high}}}$ are obtained by concatenating the outputs $\mathbf{Y}^{(q)}$ depthwise, for any $q \in \{1\,..\,Q\}$. Figure 5 shows how the above grouping is made, and how many output channels L_q each group q is assigned to.

Model	**Filt. frequency**	**Reg. param.**
WAlexNet	$[\pi/8,\ \pi/4[$	–
	$[\pi/4,\ \pi/2[$	$4.1 \cdot 10^{-3}$
	$[\pi/2,\ \pi[$	$3.2 \cdot 10^{-4}$
WResNet	any	–

Table 4. Regularization hyperparameters λ_q for each group q of DT-$\mathbb{C}$WPT feature maps. The groups with only one feature map do not need any regularization. The second and third rows of WAlexNet correspond to the blue and magenta groups in Fig. 5a, respectively.

Model	Free	Gabor	**Total**
AlexNet			
Standard	–	–	*602*
$\mathbb{R}$Max[a]	376	571	947
$\mathbb{C}$Mod[a]	376	**298**	**674**
Blur	–	–	*2 208*
Blur$\mathbb{R}$Max[a] + Blur	1 179	2 258	3 437
$\mathbb{C}$Mod[a] + Blur	1 179	**298**	**1 477**
ResNet-34			
Standard	–	–	*2 760*
$\mathbb{R}$Max[a]	1 781	1 229	3 011
$\mathbb{C}$Mod[a]	1 781	**527**	**2 308**
Blur	–	–	*3 563*
Blur$\mathbb{R}$Max[a] + Blur	2 283	1 530	3 813
$\mathbb{C}$Mod[a] + Blur	2 283	**527**	**2 810**
ABlur	–	–	*6 046*
ABlur$\mathbb{R}$Max[a] + ABlur	3 835	2 462	6 297
$\mathbb{C}$Mod[a] + ABlur	3 835	**527**	**4 362**

Table 5. Number of elements ($\times 1\,000$) in the intermediate and output feature maps during forward propagation, for a given input image of size $3 \times 224 \times 224$. Only the layers shown in Figs. 2 and 6 are taken into account.

During training, the above process aims at selecting one single DT-$\mathbb{C}$WPT feature map among each group. This is achieved through mixed-norm l^∞/l^1 regularization, as introduced in (7). The regularization hyperparameters λ_q have been chosen empirically. If they are too small, then regularization will not be effective. On the contrary, if they are too large, then the regularization term will become predominant, forcing the trainable parameter vector $\boldsymbol{\alpha}_l^{(q)}$ to randomly collapse to 0 except for one element. The chosen values of λ_q are displayed in Tab. 4.

Finally, the split L_{low}-L_{high} between the freely-trained

and Gabor channels, provided in the last row of Tab. 1, have been empirically determined from the standard models.

A.3. Kernel Visualization

The resulting convolution kernels $\mathbf{V} \in \mathcal{S}^{64\times 3}$, satisfying (1), are shown in Figs. 7 and 8 for AlexNet-based models, Figs. 9 to 11 for ResNet-based models trained on ImageNet and Figs. 12 to 14 for ResNet-based models trained on CIFAR-10. The kernels are shown as RGB color images, for both freely-trained and Gabor channels.

We can notice that, up to a few exceptions, the freely-trained channels (4 or 5 first rows) have been specialized to lower-frequency filters (mono- or bi-color blobs).

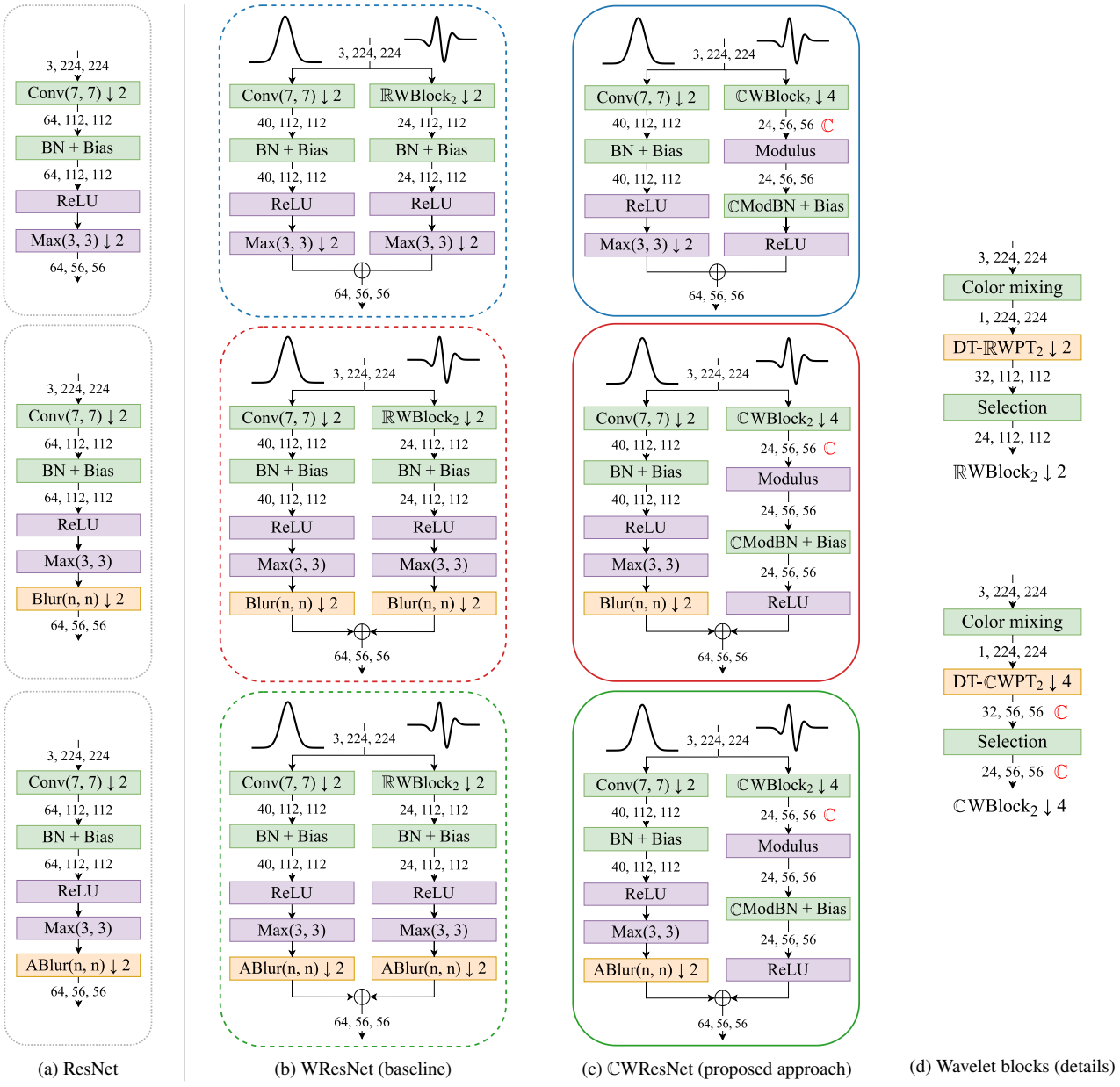
In the $\mathbb{C}$Mod-versions of our models, the 3 or 4 last rows display the complex-valued kernels $\mathbf{W} \in \mathcal{S}^{L_{\text{high}}\times 3}$ such that the outputs $\mathrm{Y}_l^{\text{mod}}$ satisfy

$$\mathrm{Y}_l^{\text{mod}} = \left| \sum_{k=1}^{K} (\mathrm{X}_k \star \mathrm{W}_{lk}) \downarrow 2^J \right|. \tag{27}$$

According to (15), we have $\mathrm{W}_{lk} = \mu_k \widetilde{\mathrm{W}}_l$, where $\widetilde{\mathrm{W}}_l$ has been introduced in (9). When looking at the power spectra, these filters appear well-localized in the Fourier domain (only one bright spot, versus two in the $\mathbb{R}$Max-based models).

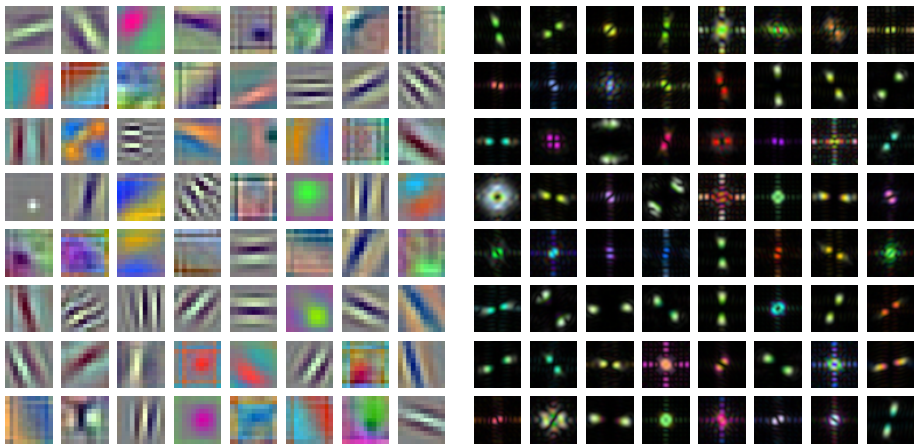
A.4. Memory Consumption

According to Sec. 4.3, our method nearly achieves, or even exceeds, the predictive power of WCNNs antialiased with blur pooling methods, with a significantly reduced memory footprint. Table 5 displays the size of intermediate and output feature maps for the layers presented in Figs. 2 and 6. We notice that replacing $\mathbb{R}\text{Max}^{\text{a}}$ by $\mathbb{C}\text{Mod}^{\text{a}}$ drastically reduces the memory consumption for the Gabor channels.

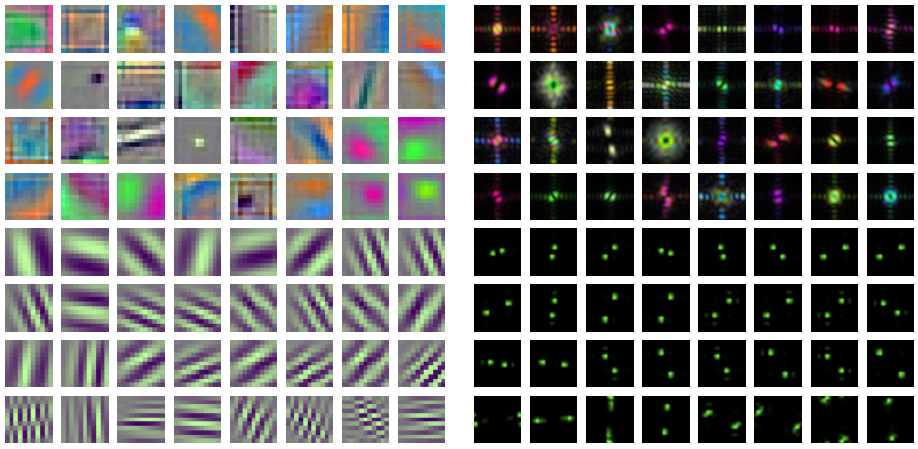


(a) ResNet (b) WResNet (baseline) (c) ℂWResNet (proposed approach) (d) Wavelet blocks (details)

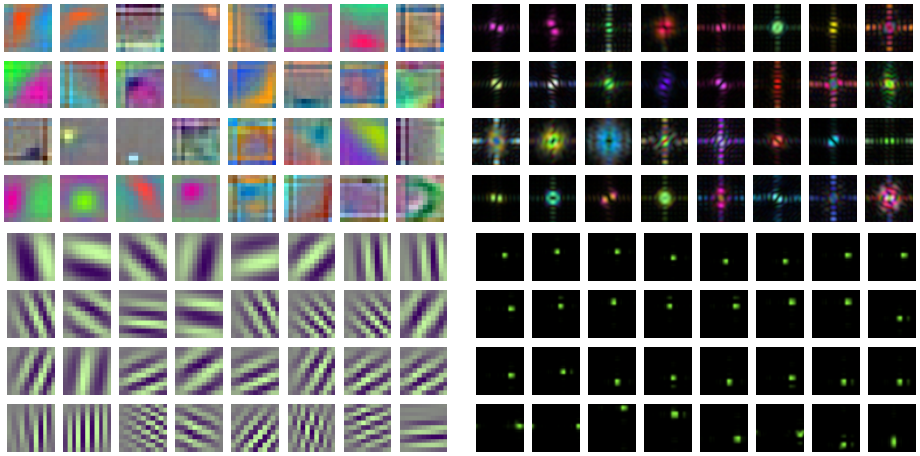
Figure 6. First layers of ResNet and its variants, corresponding to a convolution layer followed by ReLU and max pooling. The models are framed according to the same colors and line styles as in Fig. 3b. The bias modules from Fig. 2 have been replaced by an affine batch normalization layer ("BN + Bias", or "ℂModBN + Bias" when placed after Modulus—see Appendix A.1). Top: ResNet without blur pooling. Middle: Zhang's "blurpooled" models [40]. Bottom: Zou *et al.*'s approach, using adaptive blur pooling [42].



(a) Standard

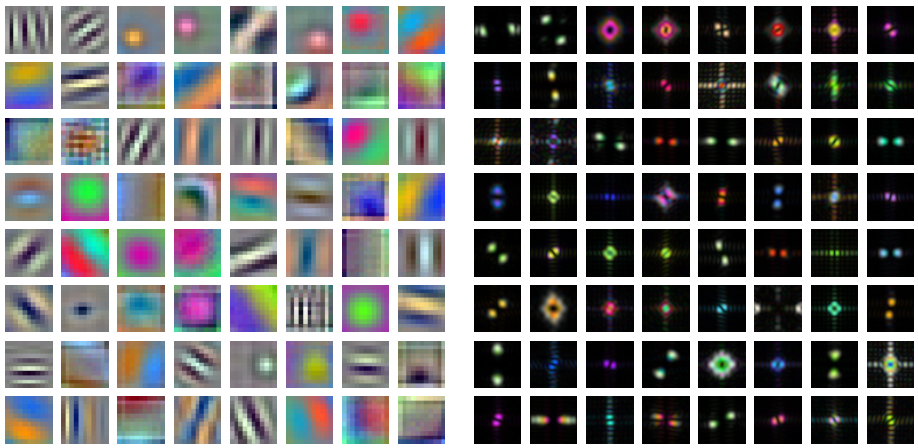


(b) $\mathbb{R}\text{Max}^{a}$ (four bottom rows)

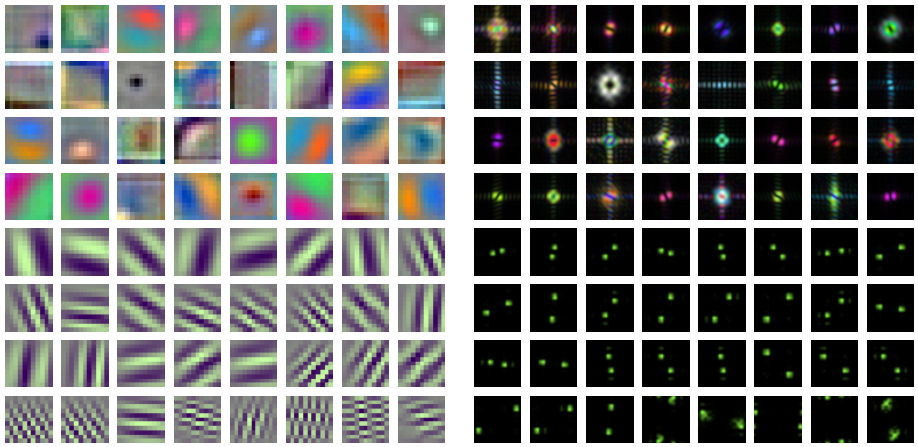


(c) $\mathbb{C}\text{Mod}^{a}$ (four bottom rows). Spatial domain: real part only

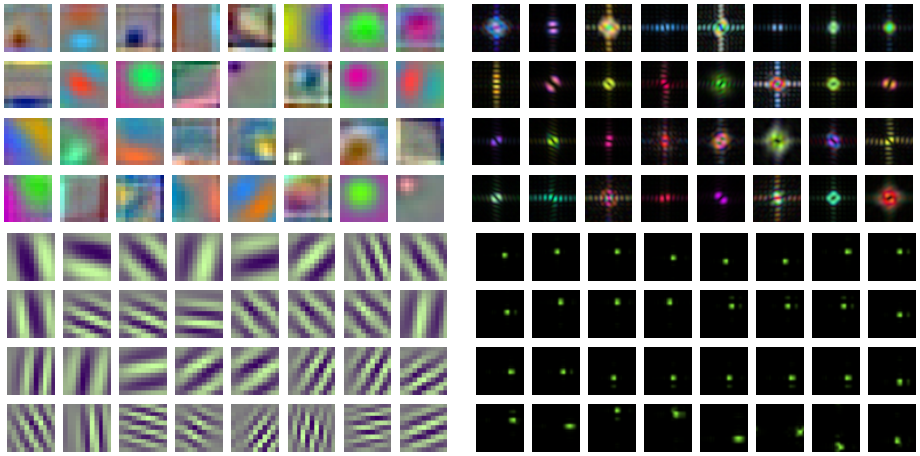
Figure 7. AlexNet (ImageNet, no blur pooling).



(a) Blur

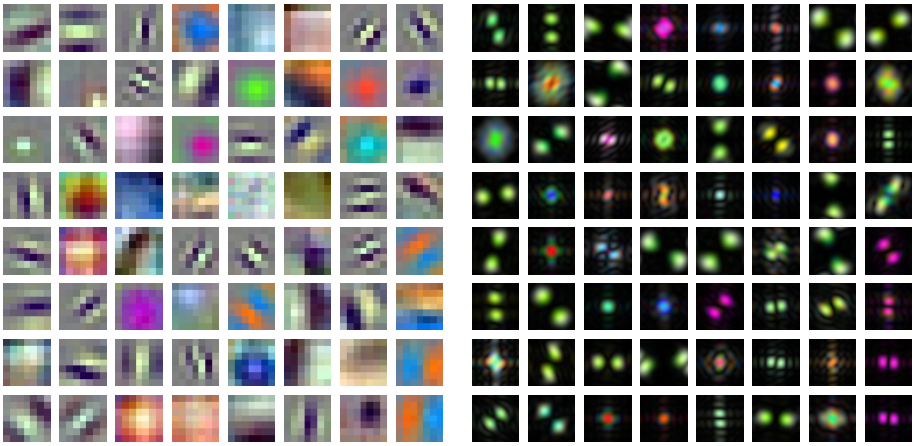


(b) Blur$\mathbb{R}$Max^{a} (four bottom rows) + Blur

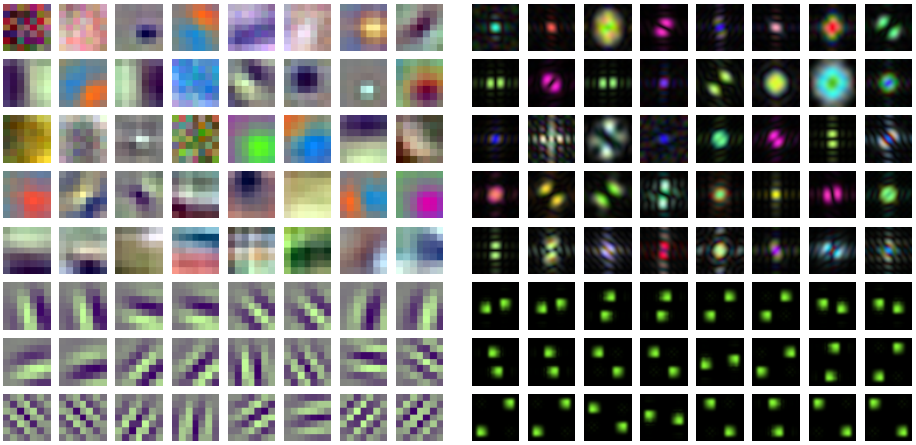


(c) $\mathbb{C}$Mod^{a} (four bottom rows) + Blur. Spatial domain: real part only

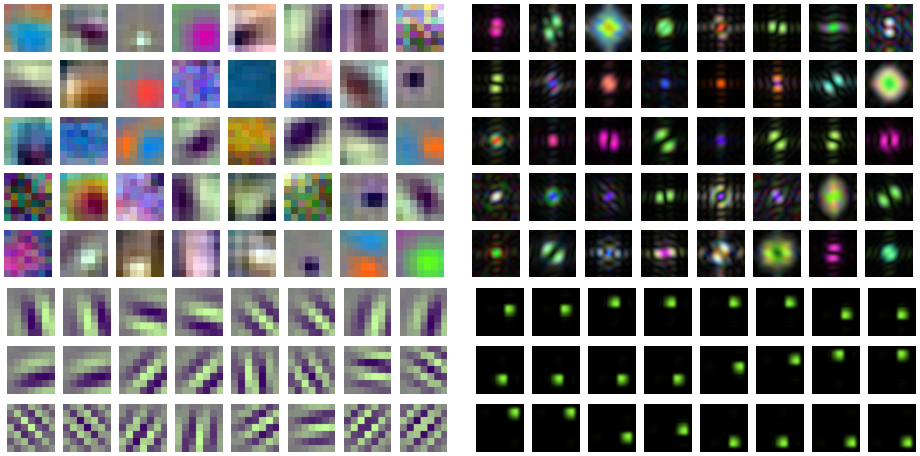
Figure 8. AlexNet with static blur pooling (ImageNet).



(a) Standard

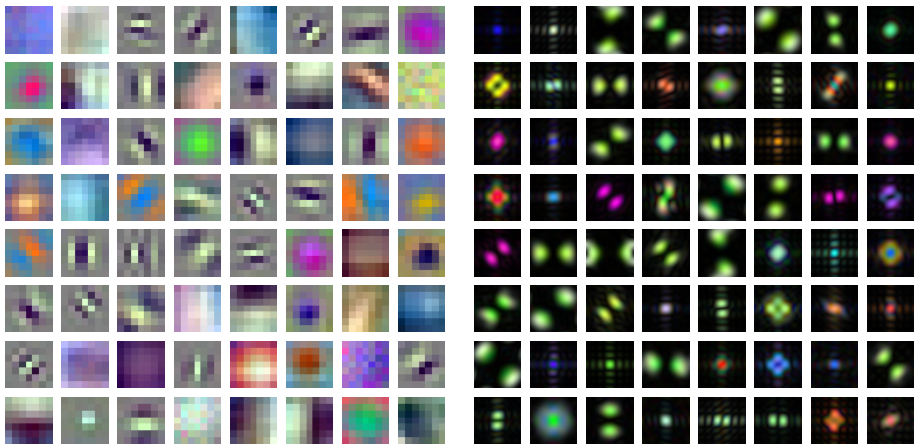


(b) $\mathbb{R}\text{Max}^{a}$ (three bottom rows)

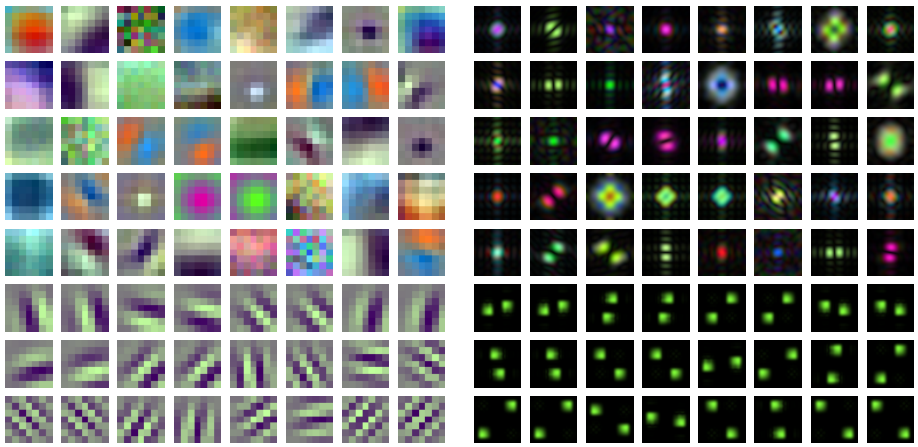


(c) $\mathbb{C}\text{Mod}^{a}$ (three bottom rows). Spatial domain: real part only

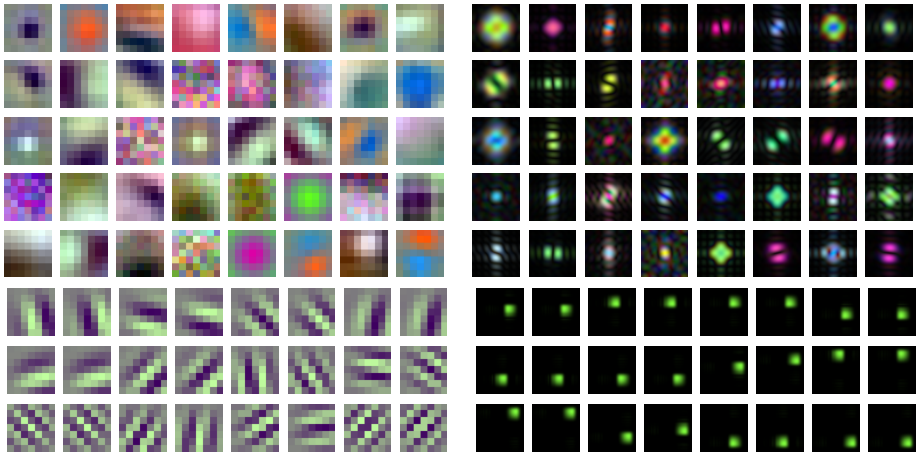
Figure 9. ResNet-34 (ImageNet, no blur pooling).



(a) Blur

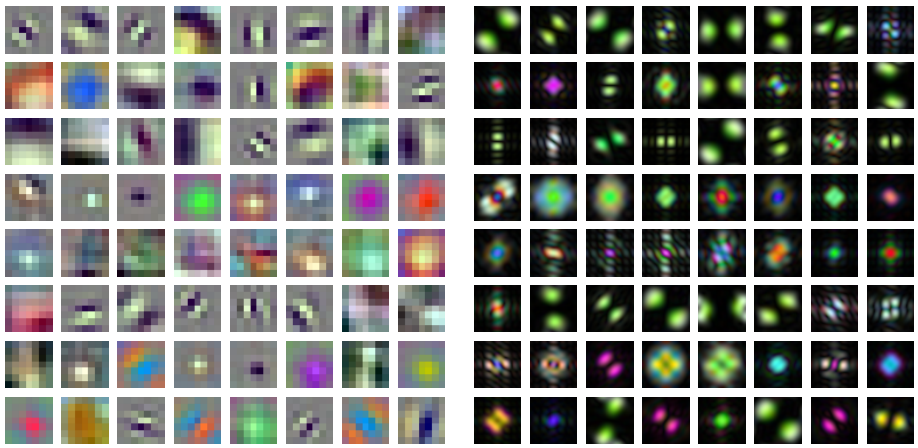


(b) Blur$\mathbb{R}$Max^{a} (three bottom rows) + Blur

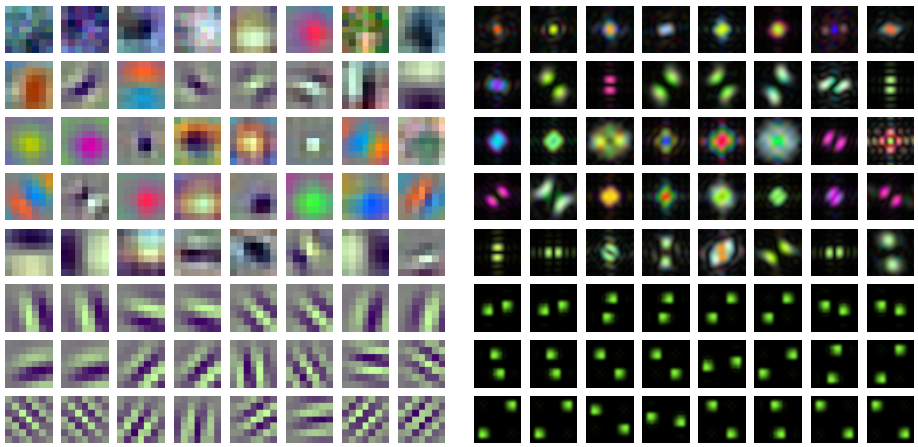


(c) $\mathbb{C}$Mod^{a} (three bottom rows) + Blur. Spatial domain: real part only

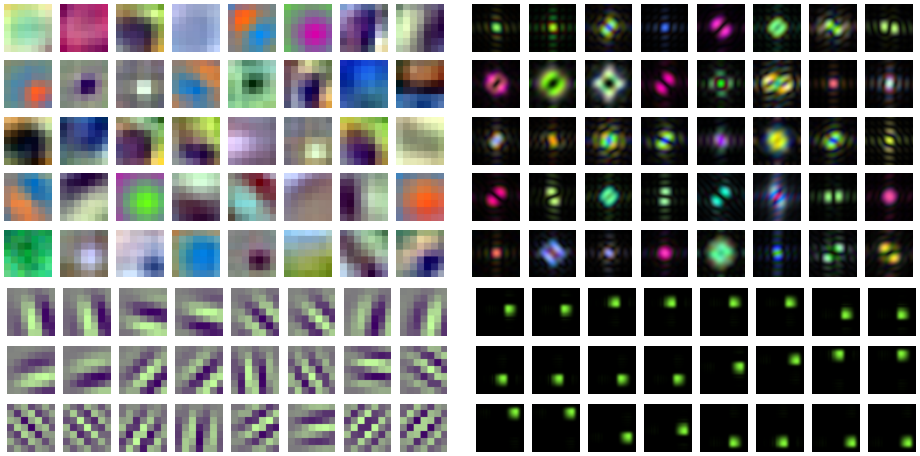
Figure 10. ResNet-34 with static blur pooling (ImageNet).



(a) ABlur

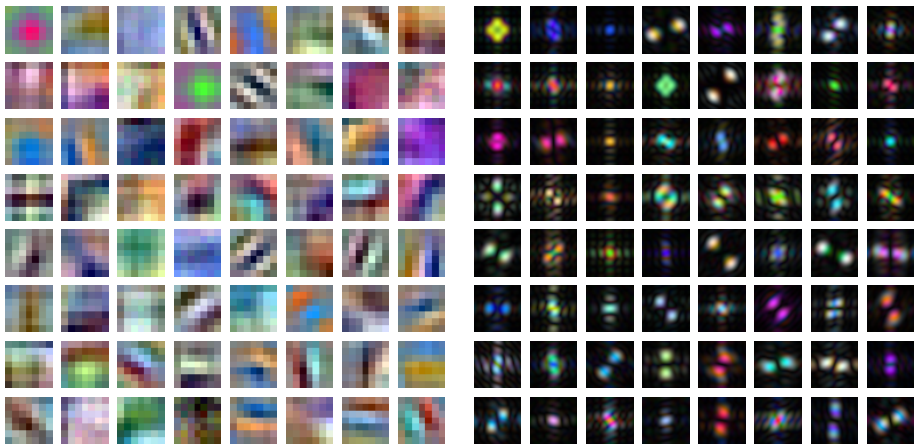


(b) ABlur$\mathbb{R}$Max^{a} (three bottom rows) + ABlur

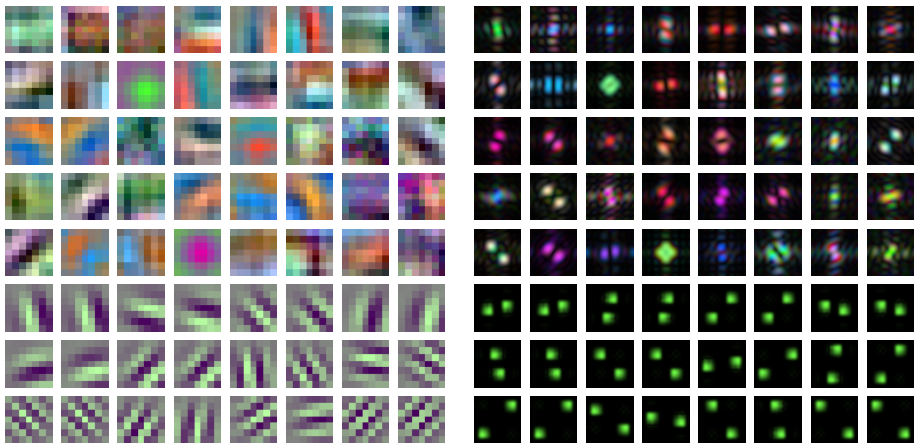


(c) $\mathbb{C}$Mod^{a} (three bottom rows) + ABlur. Spatial domain: real part only

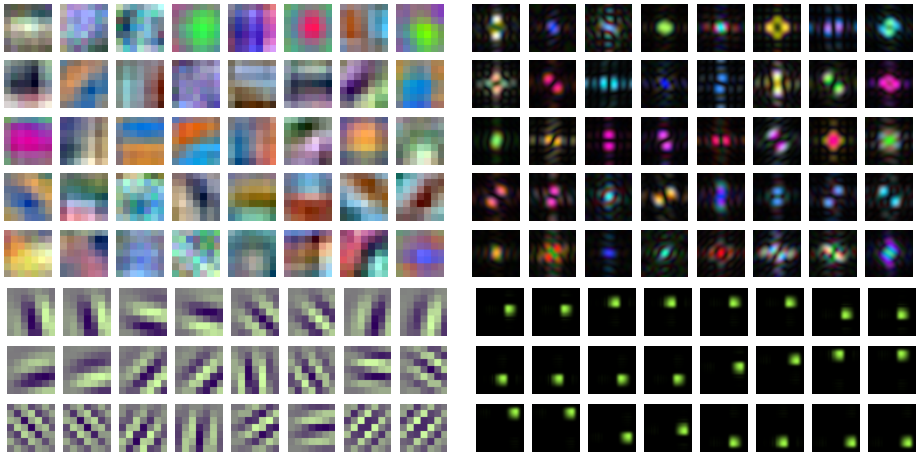
Figure 11. ResNet-34 with adaptive blur pooling (ImageNet).



(a) Standard

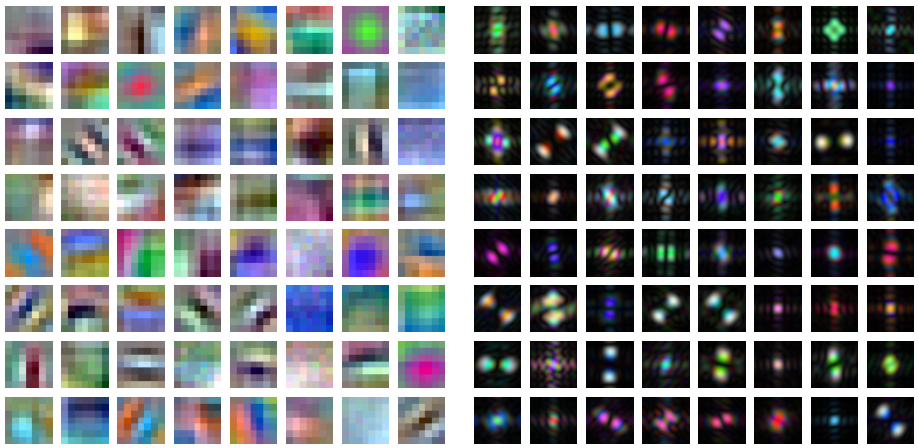


(b) $\mathbb{R}\text{Max}^a$ (three bottom rows)

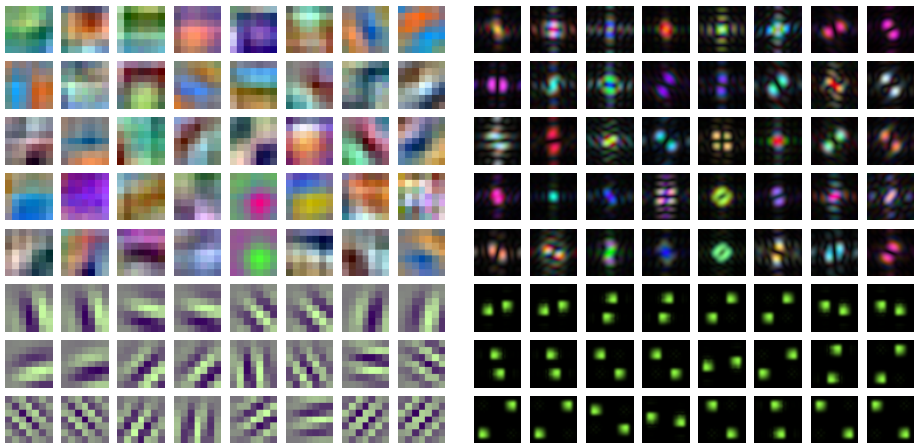


(c) $\mathbb{C}\text{Mod}^a$ (three bottom rows). Spatial domain: real part only

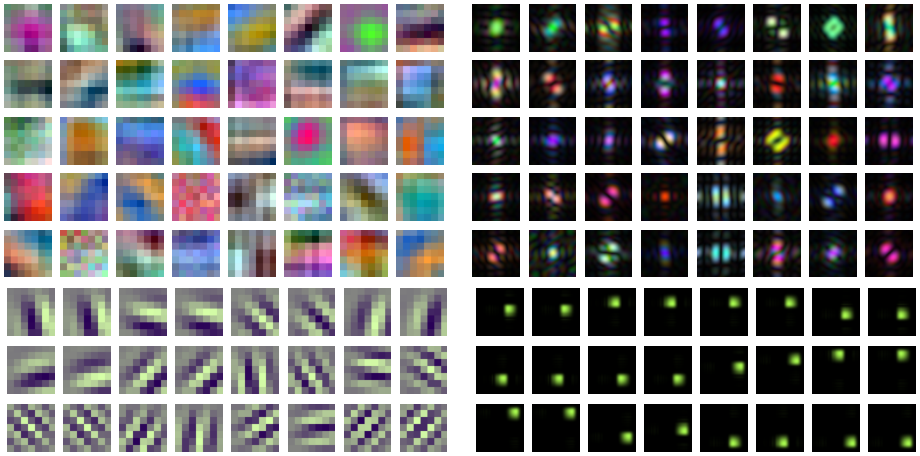
Figure 12. ResNet-18 (CIFAR-10, no blur pooling).



(a) Blur

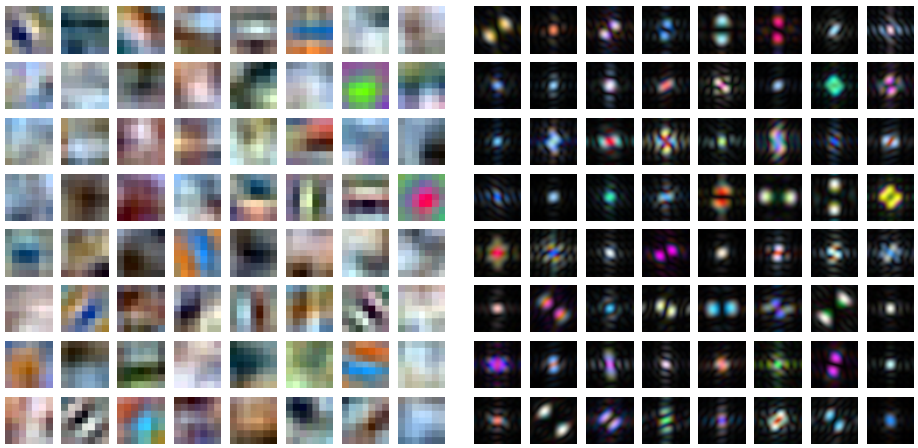


(b) Blur$\mathbb{R}$Max^{a} (three bottom rows) + Blur

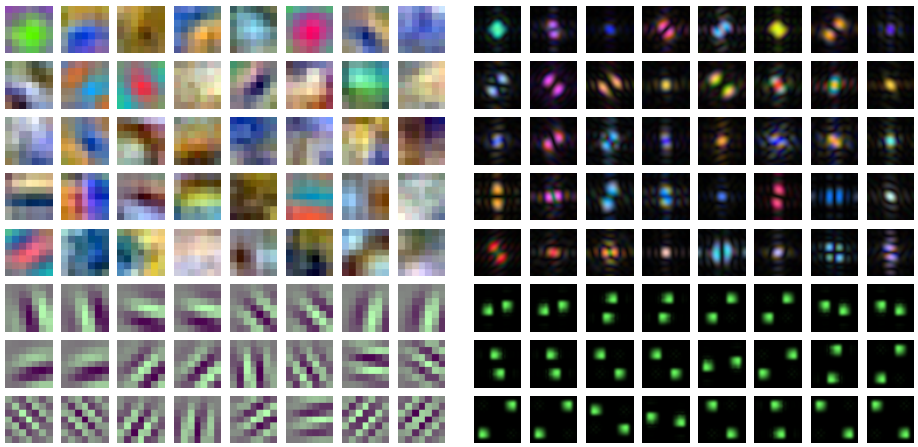


(c) $\mathbb{C}$Mod^{a} (three bottom rows) + Blur. Spatial domain: real part only

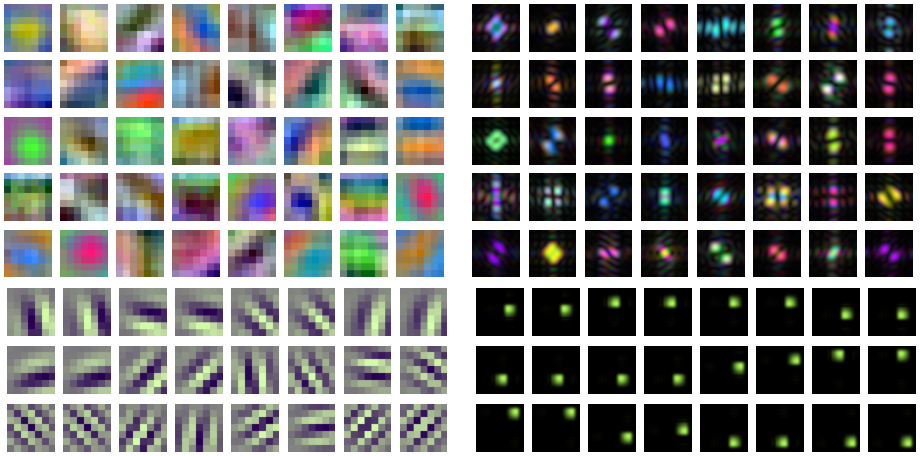
Figure 13. ResNet-18 with static blur pooling (CIFAR-10).



(a) ABlur



(b) ABlur$\mathbb{R}$Max^{a} (three bottom rows) + ABlur



(c) $\mathbb{C}$Mod^{a} (three bottom rows) + ABlur. Spatial domain: real part only

Figure 14. ResNet-18 with adaptive blur pooling (CIFAR-10).