Co-domain Symmetry for Complex-Valued Deep Learning

Utkarsh Singhal Yifei Xing Stella X. Yu
UC Berkeley / ICSI

Abstract

We study complex-valued scaling as a type of symmetry natural and unique to complex-valued measurements and representations. Deep Complex Networks (DCN) extend real-valued algebra to the complex domain without addressing complex-valued scaling. SurReal extends manifold learning to the complex plane, achieving scaling invariance with manifold distances that discard phase information.

Treating complex-valued scaling as a co-domain transformation, we design novel equivariant/invariant layer functions and architectures that exploit co-domain symmetry. We also propose novel complex-valued representations of RGB images, where complex-valued scaling indicates hue shift or correlated changes across color channels.

Benchmarked on MSTAR, CIFAR10, CIFAR100, and SVHN, our co-domain symmetric (CDS) classifiers deliver higher accuracy, better generalization, more robustness to co-domain transformations, and lower model bias and variance than DCN and SurReal with far fewer parameters.

1. Introduction

Symmetry is one of the most powerful tools in the deep learning repertoire. Naturally occurring symmetries lead to structured variation in natural data. Modeling these symmetries thus greatly simplifies learning [1], e.g., Convolutional Neural Networks (CNNs) [2] capture the *translational symmetry* of image data, and PointNet [3] captures the *permutation symmetry* of 3D point clouds. These symmetries are formalized as *invariance* or *equivariance* to a group of transformations [4]. However, this line of research has primarily focused on transformations defined on the *domain* of an image (such as scaling and rotations [5–7]), while co-domain transformations (Fig. 1) such as color shift and complex-valued range scaling remain under-explored. Additionally, this research has primarily focused on real-valued data.

We explore complex-valued data which arise naturally in **1)** remote sensing such as synthetic aperture radar (SAR) imaging, medical imaging such as magnetic resonance imaging (MRI), and radio frequency communications; **2)** spectral representations of real-valued data such as Fourier Transform [8, 9]; and **3)** physics and engineering applications [10]. In deep learning, complex-valued models have shown several benefits over their real-valued counterparts: larger representational capacity [11], more robust embedding [12] and associative memory [13], more efficient multi-task learning [14], and higher quality MRI image reconstruction [15]. We approach complex-valued deep learning from a symmetry perspective: *Which symmetries are inherent in complex-valued data, and how do we exploit them in modeling?*

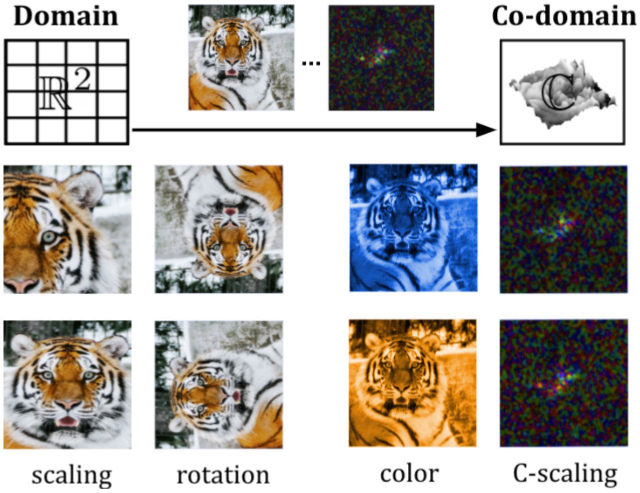
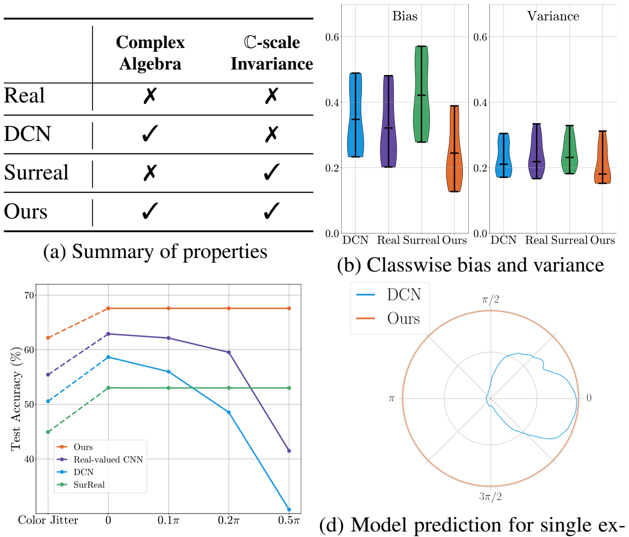


Figure 1. We study principled deep learning designs that exploit co-domain symmetry in the range of an image. An image is a function from pixel coordinates in the domain $\mathbb{R}^D$ to pixel values in the co-domain $\mathbb{C}^K$ (e.g., $(D, K) = (2, 3)$ for RGB images). Spatial transformations such as scaling and rotations act on the domain, mapping points in $\mathbb{R}^D$ to other points, while leaving the underlying function values intact. Co-domain transformations such as color distortion or complex-valued scaling, on the other hand, act on the function values only. Rows 2-3 in Column 4 are complex-valued scaled SAR images with magnitudes and phases visualized in the color intensity and hue respectively.

One type of symmetry inherent to complex-valued data is *complex-valued scaling ambiguity* [18]. For example, consider a complex-valued MRI or SAR signal $\mathbf{z}$. Due to the nature of signal acquisition, $\mathbf{z}$ could be subject to global magnitude scaling and phase offset represented by a complex-valued scalar s, thus becoming $s \cdot \mathbf{z}$.

A complex-valued classifier takes input $\mathbf{z}$ and ideally



	Complex Algebra	ℂ-scale Invariance
Real	✗	✗
DCN	✓	✗
Surreal	✗	✓
Ours	✓	✓

(a) Summary of properties

(b) Classwise bias and variance

(c) Color transformation robustness

(d) Model prediction for single example under ℂ-scaling

Figure 2. Our method combines the strengths of DCN and SurReal, demonstrating better generalization and increased robustness to ℂ-scaling and color transformations. All examples are from CIFAR 10 with our LAB encoding. **(a)** Unlike DCN or SurReal, our model handles two aspects essential to complex-valued data: ℂ-scale invariance and complex algebra. Our key insight is to design novel layer functions that are equivariant and invariant to complex scaling in the rich family of functions with complex algebra. **(b)** Violin plots of classwise bias/variance computed following the method of [16]. The whiskers represent maximum/median/minimum values respectively. While SurReal has the highest bias and variance, our model achieves the lowest, indicating better generalization. **(c)** Accuracy under color jitter (as used in [17]) and complex-scaling with different rotation ranges. Our method maintains high accuracy across complex-rotations and color jitter, whereas DCN and Real-valued CNN fail. SurReal [18] is robust, but has low overall accuracy. Our method combines high accuracy with robustness. **(d)** Model confidence of the correct class for a single example. Higher confidence means larger radius. DCN predictions are highly variable, while our model is robust to complex-scaling.

should focus on discriminating among instances from different classes, not on the instance-wise variation $s\cdot\mathbf{z}$ caused by complex-valued scaling. Formally, function f is called **complex-scale invariant** if $f(s\cdot\mathbf{z}) = f(\mathbf{z})$ and called **complex-scale equivariant** if $f(s\cdot\mathbf{z}) = s\cdot f(\mathbf{z})$. For brevity, we abbreviate complex-valued scaling as **ℂ-scaling**.

We distinguish two types of image transformations, viewing an image as a function defined over spatial locations. ℂ-scaling of a complex-valued image is a transformation in the co-domain of the image function, as opposed to a spatial transformation in the domain of the image (Fig. 1). Formally, $I : \mathbb{R}^D \to \mathbb{C}^K$ denotes a complex-valued image of K channels in the D-dimensional space, where $\mathbb{R}$ ($\mathbb{C}$) denotes the set of real (complex) numbers. Some common (D, K)'s are (2,1) for grayscale images, (2,3) for RGB, and $(3, 6+)$ for diffusion tensor images.

1. **Domain transformation** $T : \mathbb{R}^D \to \mathbb{R}^D$ transforms the spatial coordinates of an image, resulting in a spatially warped image $I(T(p))$, where $p \in \mathbb{R}^D$ denotes the pixel location. Translation, rotation, and scaling are examples of domain transformations.
2. **Co-domain transformation** $T' : \mathbb{C}^K \to \mathbb{C}^K$ maps the pixel value to another value, resulting in a color adjusted image $T'(I(p)), p \in \mathbb{R}^D$. ℂ-scaling and color distortions are examples of co-domain transformations.

ℂ-scaling thus presents not only a practical setting but also a case study for general co-domain transformations.

Existing methods approach complex-valued deep learning in two different ways. **1)** Deep Complex Networks (DCN) [19] extends real-valued algebra to the complex domain without addressing ℂ-scaling; their models are highly sensitive to ℂ-scaling (Figs. 2c and 8a). A pre-processing trick to remove such scaling ambiguity is to simply normalize all the pixel values by setting their average phase to 0 and magnitude to 1, but this process introduces artifacts when the phase distribution varies greatly with the content of the image (Fig. 8c). **2)** SurReal [18] extends manifold-valued deep learning to complex-valued data, achieving ℂ-scaling invariance using manifold distances. However, these manifold distances discard rich phase information, and the restrictive SurReal framework is unable to express complex algebraic operations on complex-valued data. As a result, it underperforms on large datasets (Tab. 1 and Fig. 2c).

We propose a principled method by designing novel layer functions that preserve co-domain symmetry. Our work makes the following contributions. **1)** We develop counterparts of common layer functions used in computer vision pipelines that are equivariant and invariant to ℂ-scaling. Our method circumvents the limitations of SurReal [18] and achieves high accuracy with larger models and datasets. **2)** We introduce novel complex-valued encodings of color, demonstrating the utility of using complex-valued representations for real-valued data. ℂ-scaling invariance under our *LAB* encoding automatically leads to color distortion robustness without the need for color jitter augmentation. **3)** Benchmarked on MSTAR, CIFAR 10, CIFAR 100, and SVHN, our method outperforms DCN and SurReal with higher accuracy, better generalization, and more robustness using far fewer parameters.

2. Related Work

Complex-valued processing. Complex numbers are ubiquitous in mathematics, physics, and engineering [10, 20, 21]. Traditional complex-valued data analysis involves higher-order statistics [22, 23]. [11] demonstrates higher representational capacity of complex-valued processing on the XOR problem. [24] proposes a sparse coding layer utilizing complex basis functions. [25] proposes a biologically meaningful complex-valued model. [26, 27] encode the confidence and size of pairwise affinity in complex-valued measurements and learn a global data embedding in the complex plane. [15] applies complex-valued neural networks to MRI image reconstruction. [28] investigates the role of critical points in complex neural networks. [29] demonstrates that complex-valued networks have smaller generalization errors than real-valued networks. [19] contains a detailed account of complex-valued deep learning.

Transformation equivariance and invariance. Most work focuses on developing convolutional layers equivariant to domain transformations such as rotation and scaling [5, 6, 30, 31]. [6] introduces a principled method for producing group-equivariant layers for finite groups. [5] extends this work to Lie groups on continuous data. [32] uses circular harmonics to produce deep neural networks equivariant to rotation and translation. [33] attempts to produce a general theory of group-equivariant CNNs on the Euclidean space and the sphere. [34] further extends the framework to local gauge transformations on the manifold. These methods are not applicable to the *co-domain* transformation we study here. [7] introduces rotation-equivariant layers for point-clouds, generalizing neurons to $\mathbb{R}^3$ vectors with 3D rotations as a co-domain transformation. In contrast, our method handles both the complex-valued algebra and the geometry of complex-valued scaling.

Complex-valued scaling. Despite increasing interests in complex-valued neural networks, how to handle $\mathbb{C}$-scaling ambiguity remains an open issue [19, 35–37] extend real-valued neural architectures to the complex domain by re-defining building blocks such as complex-valued convolution, batch normalization, and non-linear activation functions. However, these methods are not robust against $\mathbb{C}$-scaling. SurReal [18] achieves invariance to complex-valued scaling by adopting a manifold view of complex numbers. It models a complex number as an element of a manifold where $\mathbb{C}$-scaling corresponds to translation and uses tools from manifold-valued learning to create models invariant to $\mathbb{C}$-scaling. SurReal generalizes better to unseen complex-valued data with much leaner models. However, SurReal is highly restrictive (Sec. 3.1), and its complex-valued stages are forced to be linear (Sec. 3.2), limiting SurReal's modelling capacity and preventing it from achieving high accuracy on large datasets (Tab. 1).

3. Co-Domain Symmetric Learning

We treat complex-valued scaling as a co-domain transformation and design novel equivariant and invariant layer functions and architectures that exploit co-domain symmetry. We can divide $\mathbb{C}$-scaling into two parts: magnitude and phase. Since magnitude variations can be handled by normalizing the input, we focus primarily on building layers that are equivariant/invariant to phase scaling.

This section describe equivariant versions of convolution, non-linearity, pooling, and BatchNorm followed by invariant layers. We also describe GTReLU, a generalized version of the Tangent ReLU [18] non-linear activation function. Finally, we introduce prototype distance layers to convert complex-valued features into equivariant/invariant real-valued predictions.

3.1. Equivariant Convolution

Convolutional layers form a crucial part of modern computer vision pipelines. [19] describes a generalization of real-valued convolution to complex-valued filters and inputs. For pedagogical clarity, we summarize this construction here.

We start with a complex-valued feature $\mathbf{z} = \mathbf{x} + i\mathbf{y}$ where $\mathbf{x}$ denotes the real-valued part, $\mathbf{y}$ denotes the imaginary part, and $i = \sqrt{-1}$. Then for a complex-valued filter matrix $\mathbf{W} = \mathbf{A} + i\mathbf{B}$, [19] defines the complex-valued convolution $\mathbf{W} * \mathbf{z}$ using a combination of real-valued convolutions:

$$\mathbf{W} * \mathbf{z} = (\mathbf{A} * \mathbf{x} - \mathbf{B} * \mathbf{y}) + i(\mathbf{A} * \mathbf{y} + \mathbf{B} * \mathbf{x}) \quad (1)$$

In practice, a bias term is also added after the convolution to create an affine function. The structure of the weight matrix $\mathbf{W}$ results in translational equivariance, and works like [1, 5, 6] generalize it to transformations beyond translation.

In contrast to domain transformations which require a structured weight matrix, any linear layer is equivariant to complex-valued scaling: For a linear function $L : \mathbb{C}^m \to \mathbb{C}^n$ with an input vector $x \in \mathbb{C}^m$ and complex scalar $s \in \mathbb{C}$, $L(s \cdot x) = s \cdot L(x)$. However, the bias term used in DCN [19] destroys $\mathbb{C}$-scale equivariance. Thus, we remove this term, restoring its equivariance. Additionally, we use Gauss' multiplication trick to speed up the convolution by 25%:

$$\mathbf{W} * \mathbf{z} = (\mathbf{t_1} - \mathbf{t_2}) + i(\mathbf{t_3} - \mathbf{t_2} - \mathbf{t_1})$$

where $\mathbf{t_1} = \mathbf{A} * \mathbf{x}, \mathbf{t_2} = \mathbf{B} * \mathbf{y}, \mathbf{t_3} = (\mathbf{A} + \mathbf{B}) * (\mathbf{x} + \mathbf{y})$. In contrast, SurReal uses weighted Frechet Mean (wFM), a restricted convolution where the weights are constrained to be real-valued, positive, and to sum to 1. This restrictive definition leads to significantly lower accuracy (Tab. 4).

3.2. Equivariant Non-Linearity

Non-linear activation functions are necessary to construct deep hierarchical representations. [18, 19, 37, 38] have investigated several complex-valued non-linearities. $\mathbb{C}$ReLU, the most prominent example, computes ReLU independently on the real and imaginary parts of the input. Tangent ReLU (TReLU) [18] uses the polar representation, thresholding both magnitude and phase.

However, these non-linearities are not complex-scale equivariant. DCN [19] uses $\mathbb{C}$ReLU, failing to be robust against complex-scaling. SurReal does not use non-linearities in its complex-valued stages (see Tables I & II in [18]). SurReal's complex-valued stages are thus fully linear, significantly limiting its modelling capacity (Tab. 1).

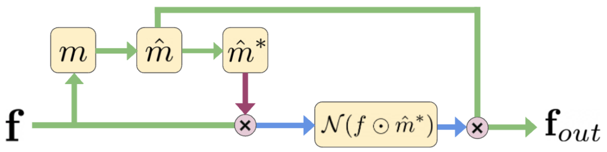


Figure 3. Our equivariant non-linearity, $\mathcal{E}\{\mathcal{N}\}$, works in four stages. We compute the channel mean m of the input feature f and normalize it to retain only phase information. This normalized mean vector $\hat{m}$ is equivariant to phase. We multiply f by the conjugate, $\hat{m}^*$, to cancel the input phase, resulting in a phase-invariant feature $f \odot \hat{m}^*$. We feed this feature to the non-linearity $\mathcal{N}$ and multiply by $\hat{m}$ to restore the removed phase. The result is equivariant in phase and also equivariant in magnitude if $\mathcal{N}$ is.

We introduce a method to make any non-linearity equivariant. Instead of applying the non-linearity to individual feature values, we apply it to the relative phase information between features (Fig. 3). Specifically, we subtract the average phase, apply the non-linearity, and add the original phase back.

Given a complex-valued input feature vector $\mathbf{f} \in \mathbb{C}^n$ with n channels, and given any complex-valued non-linearity $\mathcal{N} : \mathbb{C} \to \mathbb{C}$ such as $\mathbb{C}$ReLUor TReLU, we compute an equivariant version of $\mathcal{N}$ (denoted $\mathcal{E}\{\mathcal{N}\}$) as:

$$\mathbf{f}_{out} = \mathcal{E}\{\mathcal{N}\}(\mathbf{f}) = \hat{\mathbf{m}} \odot \mathcal{N}(\mathbf{f} \odot \hat{\mathbf{m}}^*) \tag{2}$$

where $\hat{\mathbf{m}}$ is the *normalized mean* with unit magnitude and the same phase as the mean of $\mathbf{f}$. $\hat{\mathbf{m}}^*$ denotes its complex conjugate, and $\odot$ denotes element-wise multiplication. The normalized mean $\hat{\mathbf{m}}$ is equivariant to input phase and invariant to input magnitude. As a result, the product $\mathbf{f} \odot \hat{\mathbf{m}}^*(x, y)$ is invariant to phase and equivariant to magnitude. If $\mathcal{N}$ is equivariant to magnitude (e.g., $\mathbb{C}$ReLU), the overall layer $\mathcal{E}\{\mathcal{N}\}$ is equivariant to both phase and magnitude.

3.3. Equivariant Pooling

In real-valued networks, max-pooling selects the largest activations from a set of neighboring activations. However, for complex numbers, this method applied separately for real and imaginary channels destroys phase information and thus complex-scale equivariance. Instead, we select the pixels with the highest magnitude, preserving phase information. The result is equivariant to both magnitude and phase.

3.4. Equivariant Batch Normalization

We follow [7], computing Batch Normalization [39] only on the magnitude of each complex-valued feature, thus preserving the phase information. Given a complex-valued input feature map $\mathbf{f} \in \mathbf{C}^n$, we compute:

$$\mathbf{f}_{BN} = BN(|\mathbf{f}|) \cdot \frac{\mathbf{f}}{|\mathbf{f}| + \epsilon} \tag{3}$$

where BN refers to real-valued BatchNorm, and $\epsilon = 10^{-6}$ is an offset to ensure the normalization is numerically stable. This layer is equivariant to phase and invariant to magnitude.

3.5. Invariant Complex-Valued Invariants

In order to produce invariant complex-valued features, we introduce the **Division Layer** and the **Conjugate Multiplication Layer**. Given two complex-valued features $\mathbf{z_1}, \mathbf{z_2} \in \mathbf{C}^n$, we define:

$$\text{Div}(\mathbf{z_1}, \mathbf{z_2}) = \frac{|\mathbf{z_1}|}{|\mathbf{z_2}| + \epsilon} \exp\{i(\measuredangle \mathbf{z_1} - \measuredangle \mathbf{z_2})\} \tag{4}$$

$$\text{Conj}(\mathbf{z_1}, \mathbf{z_2}) = \mathbf{z_1}\mathbf{z_2}^* \tag{5}$$

In practice, the denominator for division can be small, so we offset the magnitude of the denominator by $\epsilon = 10^{-7}$.

While the division layer induces invariance to all complex-valued scaling, the conjugate layer only induces invariance to phase. This layer also captures some second-degree interactions similar to a bilinear layer [40]. In contrast to our layers which capture relative phase and magnitude offsets of input features, SurReal's Distance Layer achieves invariance by extracting real-valued *distances* between features, discarding detailed relative phase information in the process.

3.6. Generalized Tangent ReLU

[18] introduces Tangent ReLU, a non-linearity which thresholds phase and magnitude. For a scalar input $x \in \mathbb{C}$, TReLU is defined as:

$$\text{TReLU}(\mathbf{x}) = \max(1, |\mathbf{x}|) \exp\{i(\measuredangle \mathbf{x})_+\}$$

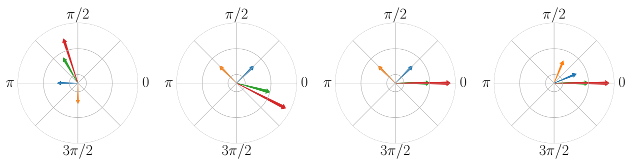
where $x_+ = \mathit{ReLU}(x) = \max(x, 0)$. In practice, TReLU slows down convergence compared to $\mathbb{C}$ReLU. We generalize TReLU through three modifications: **a)** a learned complex-valued scaling factor for each input channel, enabling the layer to adapt to input magnitude and phase, **b)** hyperparameter r to control the magnitude threshold, and **c)** learned scaling constant for the output phase of each channel, allowing the non-linearity to adapt the output phase distribution. Notably, $r = 0$ produces a phase-only version of TangentReLU, which is equivariant to input magnitude. Our proposed method generalizes TReLU both as a transformation and as a thresholding function. It is defined as:

$$\text{GTRelu}(\mathbf{x}; r, c, \omega) = \max(r, |c \cdot \mathbf{x}|) \exp\{i\omega \measuredangle (c \cdot \mathbf{x})_+\}$$

where $r \in \mathbb{C}$ is the threshold parameter, $c \in \mathbb{C}$ and $\omega \in \mathbb{R}$ are learned scaling factors (Fig. 4).

3.7. Complex Features → Real-Valued Outputs

Tasks like image classification require real-valued outputs. Complex-valued neural networks thus employ various strategies to convert complex-valued features to real-valued. SurReal [18] uses a manifold distance metric to convert pairs of complex-valued features into real-valued distances. While this approach discards rich phase information in the intermediate layers (Sec. 3.5), we note that feature distances are useful for prototype-based classification [41]. We thus propose to learn a prototype vector $\mathbf{p_i}$ for each class i and use feature distance to classify the input.



(a) Input features (b) Scaling factor (c) Thresholding (d) Phase-scaling

Figure 4. Our Generalized Tangent ReLU transforms the input in three stages: **(a)** given input complex vectors, it **(b)** multiplies each channel with a learned scaling factor, **(c)** thresholds the input magnitude and phase with hyperparameter r, and **(d)** scales the phase to adapt the output distribution.

Given a complex-valued feature vector $\mathbf{f} \in \mathbb{C}^m$, we compute the distance of $\mathbf{f}$ to every class prototype vector $\mathbf{p_i}$, and output the class with the closest prototype. Formally, the class i logit, $L_i \in \mathbb{R}$, returned by the network is defined as:

$$L_i = -\alpha \cdot d\left(\mathbf{f}, \mathbf{p_i}\right) \tag{6}$$

where α is a learned scaling factor and d is the feature distance function. As the image feature moves farther away from a class prototype, the predicted logit for that class becomes smaller. Since the features are complex-valued, a suitable metric is the manifold distance (equivalent to [18]):

$$d\left(z_1, z_2\right) = \sqrt{\left(\ln|z_1| - \ln|z_2|\right)^2 + \operatorname{arc}(\measuredangle z_1, \measuredangle z_2)^2} \tag{7}$$

where $z_1, z_2 \in \mathbb{C}$. It amplifies the effect of phase differences which would otherwise be suppressed by large variations in magnitude. Alternatively, a simple metric is the Euclidean distance. In practice, we use BatchNorm on the input features before computing distances to accelerate convergence.

Invariant classification: This layer can be made complex-scale invariant by multiplying the prototypes with an equivariant feature map:

$$L_i = -\alpha \cdot d\left(\mathbf{f}, \mathbf{p_i} \odot \bar{\mathbf{f}}\right) \tag{8}$$

where $\bar{\mathbf{f}}$ is the mean activation averaged over channels. Since both inputs to the distance function are $\mathbb{C}$-scale equivariant, the output is invariant [18].

3.8. Composing Equivariant and Invariant Layers

We introduce two patterns of model composition based on our proposed layers. **Type I** models use a complex-valued **Invariant** layer and achieve $\mathbb{C}$-scale invariance in the early stages of the model, whereas **Type E** models use **Equivariant** layers and achieve invariance in the later layers, thus retaining more phase information (Tab. 1)

Type I: These models consist of a complex-valued invariant layer (Division/Conjugate) to achieve early invariance, producing $\mathbb{C}$-scale invariant features which can be used by later stages without any architectural restrictions.

Type E: These models rely on equivariant layers, preserving the phase information through equivariant layers, and achieving late invariance. They typically achieve higher accuracy (Tab. 1), but this model class is more restrictive.

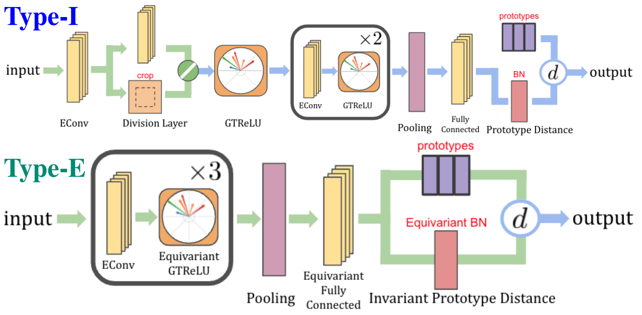


Figure 5. Our CIFARnet models demonstrate two methods of constructing complex-scale invariant models. Green arrows represent equivariant features, and blue arrows represent invariant features. **top: Type I** architecture uses a *Division Layer* in early stages, achieving early invariance. The resulting complex-scale invariant features can be used with any subsequent layers. **bottom: Type E** uses equivariant layers throughout the network, retaining phase information until the final *Invariant Prototype Distance Layer*. This class of models is more restrictive but can achieve higher accuracy (See Tab. 1) as it retains more information.

4. Complex-Valued Color Encodings

In this section, we explore complex-valued representations of real-valued image data. One such representation is the Fourier Transform, which have proven vital for signal processing applications. However, Fourier data is not spatially homogeneous or translation invariant, making it challenging for convolutional neural networks. To demonstrate the utility of our method for real-valued images, we instead propose two **complex-valued color encodings** which capture hue shift and channel correlations respectively.

Our first so-called "Sliding" encoding takes an $[R, G, B]$ image and encodes it with two complex-valued channels:

$$[R, G, B] \rightarrow [R + iG, G + iB] \tag{9}$$

The complex phase in this encoding corresponds to the ratio of R, G, B values, thus capturing the correlation between the adjacent color channels.

Our second proposed encoding uses $L^*a^*b^*$, a perceptually uniform color representation with luminance represented by the L channel and chromaticity by the a and b channels. [42] uses this color space for image colorization. We use it to represent color as a two-channel, complex-valued representation, with the first channel containing the luminance (L^* channel), and the second channel containing chromaticity (a^* and b^* channels) as $a^* + i\, b^*$ (Fig. 6):

$$[R, G, B] \rightarrow [L^*, a^* + ib^*] \tag{10}$$

Color distortions as co-domain transformations: $\mathbb{C}$-scaling in our LAB color representation (Fig. 6) approximates color distortion. Invariant models are thus naturally robust to color distortion without any data augmentation.

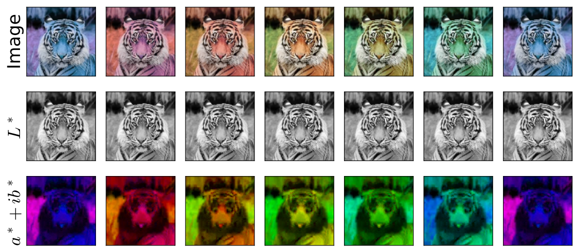


Figure 6. Visualization of our complex-valued embedding of LAB information undergoing complex-scaling. The L^* channel is visualized as a grayscale image, and the complex-valued $a^* + ib^*$ is visualized as a color image. For an image encoded with our proposed LAB encoding, color distortions can be approximated with $\mathbb{C}$-scaling.

5. Experiments

We conduct three kinds of experiments: **Accuracy**: **1)** Classification of naturally complex-valued images, **2)** real-valued images with real and complex representations; **Robustness** against complex scaling and color distortion; **Generalization: 1)** Bias-variance analysis, **2)** generalization on smaller training sets, **3)** Feature redundancy analysis.

5.1. Complex-Valued Dataset: MSTAR

MSTAR contains 15,716 complex-valued synthetic aperture radar (SAR) images divided into 11 classes [43]. Each image has one channel and size 128 × 128. We discard the last "clutter" class and follow [44], training on the depression angle 17° and testing on 15°. We train each model on varying proportions of the dataset to evaluate the accuracy and generalization capabilities of each model.

SurReal: We replicate the architecture described in Table 1 of [18]. Since the paper does not mention the learning rate, we use the same learning rate and batch size as our model. **DCN**: We use author-provided code[1], creating a complex ResNet with $\mathbb{C}$ReLU and 10 blocks per stage. By default, this model accepts 32 × 32 images, so we append $2 \times [\mathit{ComplexConv}, \mathit{ComplexBatchNorm}]$ with stride 2 to downsample the input. The model is trained for 200 epochs using SGD with batch size 64 and the learning rate schedule in [19]. We select the epoch with the best validation accuracy. **Real-valued baseline**: We use a 3-stage ResNet with 3 layers per residual block and convert the complex input into two real-valued channels.

CDS: We use a *Type I* model based on SurReal [18]. We extract equivariant features using an initial equivariant block containing *EConv*, *Eq. GTReLU*, *Eq. MaxPool* layers, and then obtain complex-scale invariant features by using a *Division Layer*. These features are then fed to a real-valued ResNet. Please refer to supplementary materials for details.

Training: We optimize both SurReal and CDS models using the AdamW optimizer [45,46] with learning rate 10^{-3}, momentum $(0.9, 0.99)$, weight decay 0.1, and batch size 256 for 2.5×10^5 iterations . We validate every 1000 steps, picking the model with the best validation accuracy.

5.2. Real-valued Datasets: CIFAR10/100, SVHN

Datasets: CIFAR10 [49] (and CIFAR100) consists of 10 (100) classes containing 6000 (600) images each. Both CIFAR10 and CIFAR100 are partitioned into 50000 training images and 10000 test images. SVHN [50] consists of house number images from Google Street View, divided into 10 classes with $73,257$ training digits and $26,032$ testing digits.

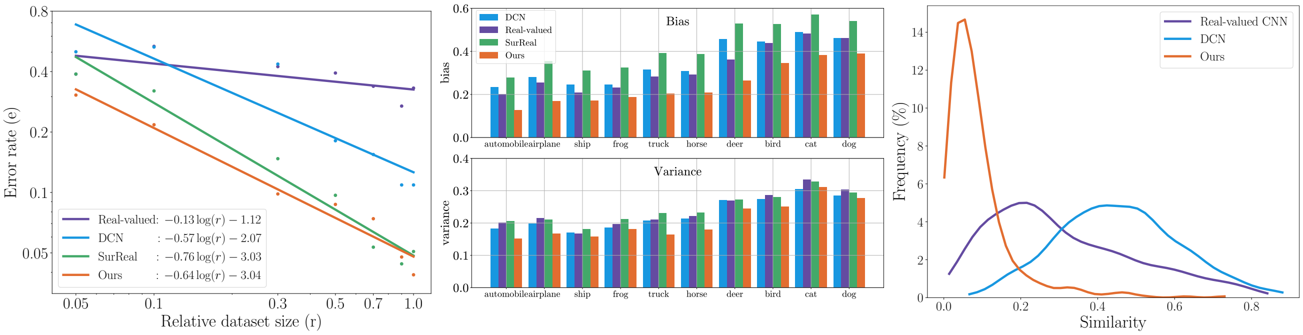
Models: To ensure equal footing for each model, all networks in this experiment are based off CIFARNet, i.e., 3 Convolution Layers (stride 2) and 2 fully connected layers. We also replace average pooling with a depthwise-separable convolution as a learnable pooling layer. All models are optimized with AdamW [45,46] using momentum $(0.99, 0.999)$, for 5×10^4 steps with batch size 256, learning rate 10^{-3}, weight decay 0.1, and validated every 1000 iterations. **DCN**: We use *ComplexConv* for convolutions and $\mathbb{C}$ReLU as the non-linearity. We do not use Residual Blocks or Complex BatchNorm from [19] to ensure fairness. **SurReal**: We use wFM for convolutions and use *Distance Transform* after Layer 3 to extract invariant real-valued features. **Real-Valued CNN**: We use the CIFARNet architecture, converting each complex input channel into two real-valued channels. **CDS**: We evaluate two models: **Type I**: We use *EConv* for convolutions and *GTReLU* $(r = 0)$ for non-linearity. We use a Division layer after the first Econv to achieve invariance. The final fully-connected layer is replaced with *Prototype Distance* layer to predict class logits (Fig. 5). **Type E**: We use *Econv* for convolutions and *Equivariant GTReLU* for non-linearity. The final FC layer is replaced with *Invariant Prototype Distance* layer to predict logits (Fig. 5), and the prototype distance inputs are normalized with *Equivariant BatchNorm* to preserve equivariance.

CDS-Large: We train a $1.7M$ parameter *Type I* model on CIFAR 10 with LAB encoding and compare it against equivalently sized DCN (WS with $\mathbb{C}$ReLU from [19]). CDS-Large is based on the simplified 4-stage ResNet provided by Page et al. [51] for DAWNBench [52]. We use the conjugate layer after the first Econv to get $\mathbb{C}$-scale invariant features and feed them to the Complex ResNet. Like DCN, we optimize the model using SGD with horizontal flipping and random cropping augmentations with a varying learning rate schedule (see supplementary material for more details).

5.3. Model Performance Analysis

Accuracy and scalability: Our approach achieves $\mathbb{C}$-scale invariance of manifold-based methods while retaining high accuracy and scalability. On **MSTAR**, our model beats

[1] https://github.com/ChihebTrabelsi/deep_complex_networks



(a) Scaling Laws for MSTAR [47] (b) Class-wise Bias and Variance for CIFAR10 [16] (c) Filter similarity histogram for *conv2*

Figure 7. Our model generalizes across various dataset sizes, has lower bias/variance, and learns diverse filters. **(a)**: We produce trend curves (similar to [47]) for the MSTAR accuracy table (Tab. 2). We use linear regression to model log error rate as a function of log dataset ratio. Our method has the lowest test error for measured dataset sizes, a trend that is predicted to scale to even smaller sizes. **(b)**: We followed [16] for CIFAR10 models with LAB encoding. Classes are ordered in ascending order of bias for our model. Our model consistently shows the lowest bias for each class, and the lowest variance for 9 out of 10 classes, indicating overall superior generalization ability. **(c)**: Filter similarity histogram from the *conv2* layer of each CIFARnet model, following [48]. Our distribution mean is closest to 0, indicating our method achieves the least redundant filters.

Method	# Params	CIFAR10			CIFAR100			SVHN		
		RGB	LAB	Sliding	RGB	LAB	Sliding	RGB	LAB	Sliding
DCN [19]	66,858	65.17	58.64	63.83	32.52	27.36	28.87	85.26	84.43	87.44
SurReal [18]	35,274	50.68	53.02	54.61	23.57	25.97	26.66	80.51	53.48	80.79
Real-valued CNN	34,282	64.43	63.00	63.43	31.93	31.72	31.93	87.47	84.93	87.37
Ours (Type-I)	24,241	**69.23**	67.17	68.7	36.92	37.81	38.51	**89.39**	**88.86**	**90.25**
Ours (Type-E)	23,697	68.48	**67.58**	**69.19**	**41.83**	**39.55**	**42.08**	77.19	74.21	88.39

Table 1. Our models outperform the baselines' CIFARnet versions on real-valued datasets. The *Type-I* model performs best on easy datasets (e.g. SVHN), and the *Type-E* model performs better on difficult datasets (e.g. CIFAR100). In contrast, SurReal is worse on all datasets.

Model	Params	5%	10%	50%	90%	100%
Real	33,050	47.4	46.6	60.6	73	66.9
SurReal [18]	63,690	61.1	68.0	90.3	**95.6**	94.9
DCN [19]	863,587	49.8	47.0	81.9	89.1	89.1
Ours	29,536	**69.5**	**78.3**	**91.3**	95.2	**96.1**

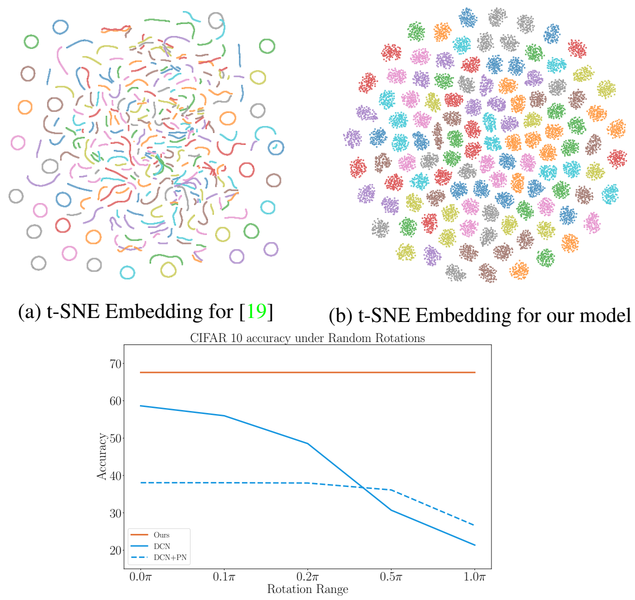
Table 2. Our method achieves the best accuracy and generalization with the fewest parameters. We report accuracy on varying proportions of MSTAR training data. The performance gap is wider for smaller train-sets, with Real-CNN and DCN failing to generalize.

the baselines across a diverse range of splits with less than half the parameters used by SurReal (Tab. 2). On the smallest training split (5% training data), our model shows a gain of 19.7% against DCN and real-valued CNN and 8.4% against SurReal. On the largest split (100%), our model beats real-valued CNN by 29.2%, DCN by 7%, and SurReal by 1.2%, showing our advantage on a large range of dataset sizes.

On **CIFAR10, CIFAR100, and SVHN** under different encodings, our models obtain the highest accuracy across every setting (Tab. 1). Unlike SurReal, our model scales to these large classification datasets while retaining $\mathbb{C}$-scale invariance. For the complex-valued color encodings, which require precise processing of phase information, our model consistently beats baselines by 4%-8%. These results highlight the advantage of our approach for precise complex-valued processing across a variety of real-valued datasets.

Phase normalization and color jitter: A natural preprocessing trick to address $\mathbb{C}$-scale invariance is to compute the average input phase $\hat{\phi}$ and to scale the input by $e^{-i\hat{\phi}}$ to cancel it. We test this approach by applying random $\mathbb{C}$-scaling with different rotation ranges and comparing DCN's accuracy with and without phase normalization against our method (Type-E) (Fig. 8c). When the input phase distribution is simple (e.g., phase set to 0), phase normalization successfully protects DCN against $\mathbb{C}$-scaling. However, for complicated phase distributions such as LAB encoding, this method fails. Our method succeeds in both situations, and this robustness transfers to the color jitter (as used by [17], see Fig. 2c). Our model is robust without data augmentation.

Bias and variance analysis: While model accuracy across different datasets is useful, a better measure for the generalization of supervised models is the bias-variance decomposition. We follow [16]: given model f, dataset D, ground truth Y, and instance x, [16] defines the bias-variance



(a) t-SNE Embedding for [19]

(b) t-SNE Embedding for our model

(c) Comparison against phase normalization

Figure 8. Our method learns invariant features with respect to complex-scaling of the input. All examples are from CIFAR 10 with our LAB encoding, undergoing multiplication by a unit complex number. **(a, b)** tSNE embedding trajectories from DCN [19] and our model. Each color represents a different example. Embeddings form tight clusters for our model, and irregular overlapping curves for DCN. **(c)** Average accuracy under different rotation ranges, comparing DCN with phase normalization (dotted blue line) and without phase normalization (solid blue line) against our method. The color encoding has a complicated phase distribution, and phase normalization fails to estimate the amount of rotation, resulting in poor accuracy. In contrast, our model is robust to $\mathbb{C}$-scaling.

decomposition of the prediction error (per instance) as:

$$\mathrm{Err}(x;f) = E\left[(f(x;D)-Y)^2\right] \tag{11}$$

$$= \mathrm{Bias}(x;f) + \mathrm{Var}(x;f) + \mathrm{Irred.\ Err}(x) \tag{12}$$

where bias measures the accuracy of the predictions with respect to the ground-truth and variance measures the stability of the predictions. Using the 0-1 loss $\mathcal{L}_{0-1}$, [16] calculates the bias and variance terms (per instance per model) for the classification task as such:

$$\mathrm{Bias}(x;h) = \mathcal{L}_{0-1}\left(y_m;t\right) \tag{13}$$

$$\mathrm{Var}(x;h) = \frac{1}{n}\sum_{k=1}^{n}\mathcal{L}_{0-1}\left(y^{(k)};y_m\right) \tag{14}$$

where y_m is the *mode* of all predictions. We compute this metric for each instance, averaging bias and variance over classes. Compared on CIFAR10 with LAB encoding, our model (Type-E) achieves the lowest bias among all classes and the lowest variance among 9 out of 10 classes (Fig. 7). In contrast, SurReal achieves significantly higher bias and variance despite being $\mathbb{C}$-scale invariant.

Method	# Params	%Acc
DCN [19]	1.7M	92.8
Ours	1.7M	**93.7**

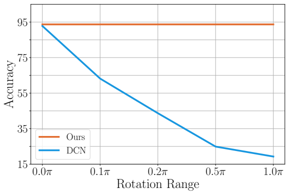


Table 3. Our model beats DCN while additionally achieving complex-scale invariance. **a)** We train DCN and CDS on CIFAR10 with the LAB encoding, achieving higher accuracy. This result is consistent with (Tab. 1), with smaller margin of improvement due to the larger capacity of big models. **b)** Similar to Figure 1g, we plot average accuracy under different rotation ranges. DCN accuracy degrades under C-scaling, while our method is robust.

Generalization from less training data: [47] derives empirical trends for scaling of language models under different conditions, including the overfitting regime where the training set is small compared to parameters. We produce similar trend curves for the MSTAR test results by fitting linear regression curves to log accuracy and dataset size reported in Tab. 2. We plot the results in Fig. 7. The extrapolated least-squares linear fit suggests our model might continue to generalize better on yet smaller datasets.

Feature redundancy comparison: [48] shows that common CNN architectures learn highly correlated filters. This increases model size and reduces the ability to capture diversity. We follow [48], measuring correlations between guided backpropagation maps of different filters in layer 2 for each model on CIFAR10 with the LAB encoding. We find that our model displays the highest filter diversity. This observation is consistent with higher test accuracy, lower bias and variance, and leaner models from previous experiments.

Scaling to large models: While edge computing and low-energy applications benefit from leaner models (Tabs. 1 and 2), state-of-the-art models on large real-valued datasets [53] contain millions of parameters. We test the scalability of our approach by comparing CDS-Large with a DCN model of equivalent size on CIFAR10 (LAB) (Tab. 3). While we focus on leaner $\mathbb{C}$-scale invariant models, our method beats DCN while additionally achieving complex-scale invariance even for large models. This observation is consistent with our results for small models (Tab. 1), showing the effectiveness of our method for diverse model sizes.

Summary: We analyze $\mathbb{C}$-scaling as a co-domain transformation and derive equivariant/invariant versions of commonly used layers. We also present novel complex encodings. Our approach combines complex-valued algebra with complex-scaling geometry, resulting in leaner and more robust models with better accuracy and generalization.

Acknowledgements: We thank DARPA, AFRL, NGA, Berkeley Deep Drive, and Berkeley AI Research/Google for funding our line of research and its applications. We also thank Matthew Tancik, Ren Ng, Connelly Barnes, and Claudia Tischler for their thoughtful comments.

References

[1] Maurice Weiler, Fred A. Hamprecht, and Martin Storath. Learning steerable filters for rotation equivariant cnns. In *CVPR*, 2018. 1, 3

[2] 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. 1

[3] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation, 2017. 1

[4] Ethan Eade. Lie groups for computer vision. *Cambridge Univ., Cam-bridge, UK, Tech. Rep*, 2, 2014. 1

[5] Marc Finzi, Samuel Stanton, Pavel Izmailov, and Andrew Gordon Wilson. Generalizing convolutional neural networks for equivariance to lie groups on arbitrary continuous data. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3165–3176. PMLR, 13–18 Jul 2020. 1, 3

[6] Taco Cohen and Max Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999. PMLR, 2016. 1, 3

[7] Congyue Deng, Or Litany, Yueqi Duan, Adrien Poulenard, Andrea Tagliasacchi, and Leonidas J. Guibas. Vector neurons: A general framework for so(3)-equivariant networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12200–12209, October 2021. 1, 3, 4

[8] Stéphane Mallat. Understanding Deep Convolutional Networks. *Philosophical Transactions A*, 374:20150203, 2016. 1

[9] Risi Kondor and Shubhendu Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. *arXiv preprint arXiv:1802.03690*, 2018. 1

[10] Tristan Needham. *Visual complex analysis*. Oxford University Press, 1998. 1, 2

[11] Tohru Nitta. The computational power of complex-valued neuron. In *Joint International Conference ICANN/ICONIP*, 2003. 1, 2

[12] Stella Yu. Angular embedding: A robust quadratic criterion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(1):158–173, 2012. 1

[13] Ivo Danihelka, Greg Wayne, Benigno Uria, Nal Kalchbrenner, and Alex Graves. Associative long short-term memory. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1986–1994, New York, New York, USA, 20–22 Jun 2016. PMLR. 1

[14] Brian Cheung, Alexander Terekhov, Yubei Chen, Pulkit Agrawal, and Bruno Olshausen. Superposition of many models into one. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 1

[15] Muneer Ahmad Dedmari, Sailesh Conjeti, Santiago Estrada, Phillip Ehses, Tony Stöcker, and Martin Reuter. Complex fully convolutional neural networks for mr image reconstruction. In Florian Knoll, Andreas Maier, and Daniel Rueckert, editors, *Machine Learning for Medical Image Reconstruction*, pages 30–38, Cham, 2018. Springer International Publishing. 1, 3

[16] Xudong Wang, Long Lian, Zhongqi Miao, Ziwei Liu, and Stella Yu. Long-tailed recognition by routing diverse distribution-aware experts. In *International Conference on Learning Representations*, 2021. 2, 7, 8, 12

[17] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, pages 1195–1204. Curran Associates, Inc., 2017. 2, 7

[18] Rudrasis Chakraborty, Yifei Xing, and Stella Yu. Surreal: Complex-valued deep learning as principled transformations on a rotational lie group. *arXiv preprint arXiv:1910.11334*, 2019. 1, 2, 3, 4, 5, 6, 7, 12

[19] Chiheb Trabelsi, Olexa Bilaniuk, Ying Zhang, Dmitriy Serdyuk, Sandeep Subramanian, João Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio, and Christopher J Pal. Deep complex networks. *arXiv preprint arXiv:1705.09792*, 2017. 2, 3, 6, 7, 8, 12

[20] Alan V Oppenheim. *Discrete-time signal processing*. Pearson Education India, 1999. 2

[21] Jon Mathews and Robert Lee Walker. *Mathematical methods of physics*, volume 501. WA Benjamin New York, 1970. 2

[22] Witold Kinsner and Warren Grieder. Amplification of signal features using variance fractal dimension trajectory. *Int. J. Cogn. Inform. Nat. Intell.*, pages 1–17, 2010. 2

[23] Juergen Reichert. Automatic classification of communication signals using higher order statistics. In *[Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 5, pages 221–224. IEEE, 1992. 2

[24] Charles F Cadieu and Bruno A Olshausen. Learning intermediate-level representations of form and motion from natural movies. *Neural computation*, 24(4):827–866, 2012. 2

[25] David P Reichert and Thomas Serre. Neuronal synchrony in complex-valued deep networks. *arXiv preprint arXiv:1312.6115*, 2013. 2

[26] Stella X. Yu. Angular embedding: from jarring intensity differences to perceived luminance. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2302–9, 2009. 3

[27] Stella X. Yu. Angular embedding: A robust quadratic criterion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(1):158–73, 2012. 3

[28] T Nitta. On the critical points of the complex-valued neural network. In *Proceedings of the 9th International Conference on Neural Information Processing, 2002. ICONIP'02.*, volume 3, pages 1099–1103. IEEE, 2002. 3

[29] Akira Hirose and Shotaro Yoshida. Generalization characteristics of complex-valued feedforward neural networks in relation to signal coherence. *IEEE Transactions on Neural Networks and learning systems*, 23(4):541–551, 2012. 3

[30] Daniel E Worrall and Max Welling. Deep scale-spaces: Equivariance over scale. *arXiv preprint arXiv:1905.11697*, 2019. 3

[31] Diego Marcos, Michele Volpi, Nikos Komodakis, and Devis Tuia. Rotation equivariant vector field networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5048–5057, 2017. 3

[32] 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*, pages 5028–5037, 2017. 3

[33] Taco Cohen, Mario Geiger, and Maurice Weiler. A general theory of equivariant cnns on homogeneous spaces. *arXiv preprint arXiv:1811.02017*, 2018. 3

[34] Taco Cohen, Maurice Weiler, Berkay Kicanaoglu, and Max Welling. Gauge equivariant convolutional networks and the icosahedral cnn. In *International Conference on Machine Learning*, pages 1321–1330. PMLR, 2019. 3

[35] Zhimian Zhang, Haipeng Wang, Feng Xu, and Ya-Qiu Jin. Complex-valued convolutional neural network and its application in polarimetric sar image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(12):7177–7188, 2017. 3

[36] Patrick Virtue, Stella X. Yu, and Michael Lustig. Better than real: Complex-valued neural networks for mri fingerprinting. In *International Conference on Image Processing*, 2017. 3

[37] Patrick Virtue. *Complex-valued Deep Learning with Applications to Magnetic Resonance Image Synthesis*. PhD thesis, UC Berkeley, 2018. 3

[38] Simone Scardapane, Steven Van Vaerenbergh, Amir Hussain, and Aurelio Uncini. Complex-valued neural networks with nonparametric activation functions. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 4(2):140–150, 2020. 3

[39] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France, 07–09 Jul 2015. PMLR. 4

[40] Yang Gao, Oscar Beijbom, Ning Zhang, and Trevor Darrell. Compact bilinear pooling. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 317–326, 2016. 4

[41] Michael Biehl, Barbara Hammer, and Thomas Villmann. Distance measures for prototype based classification. In Lucio Grandinetti, Thomas Lippert, and Nicolai Petkov, editors, *Brain-Inspired Computing*, pages 100–116, Cham, 2014. Springer International Publishing. 4

[42] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *ECCV*, 2016. 5

[43] Eric R Keydel, Shung Wu Lee, and John T Moore. Mstar extended operating conditions: A tutorial. In *Algorithms for Synthetic Aperture Radar Imagery III*, volume 2757, pages 228–242. International Society for Optics and Photonics, 1996. 6

[44] Jiayun Wang, Patrick Virtue, and Stella Yu. Successive embedding and classification loss for aerial image classification, 2019. 6

[45] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2017. 6

[46] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 6

[47] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. 7, 8

[48] Xudong Wang and Stella X Yu. Tied block convolution: Leaner and better cnns with shared thinner filters. *arXiv preprint arXiv:2009.12021*, 2020. 7, 8

[49] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6

[50] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011. 6

[51] David Page. How to train your resnet. *https://github.com/davidcpage/cifar10-fast*, 2020. 6

[52] Cody A. Coleman, Deepak Narayanan, Daniel Kang, Tian Zhao, Jian Zhang, Luigi Nardi, Peter Bailis, Kunle Olukotun, Chris Re, and Matei Zaharia. Dawnbench: An end-to-end deep learning benchmark and competition. In *NeurIPS ML Systems Workshop*, 2017. 6

[53] 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. 8