EMPOWERING NETWORKS WITH SCALE AND ROTATION EQUIVARIANCE USING A SIMILARITY CONVOLUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

The translational equivariant nature of CNN is a reason for its great success in the field of computer vision. However, networks do not enjoy more general equivariance properties such as rotation or scaling. This limits the generalization performance of the network. In this paper, we devise a method that provides networks with equivariance with respect to translation, rotation, and scaling simultaneously. We define a convolution-like operation and ensure equivariance based on our proposed scalable Fourier-Argand representation. The method has similar efficiency as a traditional network and hardly introduces any additional learnable parameters, since it does not face the computational issue often occurs in group-convolution operator. We verified the quality of our approach in the image classification task, demonstrating the robustness and the generalization ability to both scaled and rotated inputs.

1 INTRODUCTION

The great success of network architectures can be attributed to large datasets and large number of parameters, making it possible to "remember" more information. On the contrary, humans can learn new concepts with very little data, and are able to generalize this knowledge. Due to a lack of modelization of geometric deformations, networks prefer to "remember" all the data through filter parameters instead of "learning" a full generalization ability. For example, in the classification task, networks trained with datasets with specific object size will make the test fails when using the same object but with a size that does not appear in the training set. The ability to factor out transformations, such as rotation or scaling, in the learning process remains to be addressed. It is indeed quite frequent to deal with images in which objects have a different orientation and scale than in the training set, for instance, as a result of distance and orientation change of the camera.

To mitigate this issue, data augmentation before training (Krizhevsky et al., 2012) is quite common. However, this leads to a substantially larger dataset and makes training more complicated. Moreover, this strategy tends to learn a group of duplicates of almost the same filters, which usually requires more learnable parameters to achieve competitive performance. A visualization of the weights of the first layer (Zeiler & Fergus, 2014) points out that many filters are similar but rotated and scaled versions of a similar prototype, which results in significantly more redundancy.

The concept of equivariance was brought to solve this issue, which can be roughly described as follows: If the input undergoes a particular geometric transformation, the output feature from the network (with randomly initialized weights) should exhibit a similarly predictable geometric transformation. Should a network satisfy equivariance to scalings and rotations, training it with only one size and orientation would naturally generalize its performance to all sizes and orientations.

To achieve this property, methods that use group convolution have dominated this field. An over-simplified interpretation of a typical group convolution method is as follow: Features are convolved with dilated filters of the same template to obtain multi-channel features. And the distortion of the input features then corresponds to the cycle shift between channels. For example, equivariant CNNs on truncated directions (Cohen & Welling, 2016; Zhou et al., 2017) use several directional filters to obtain equivariance within a discrete group. Further works generalized the rotation equivariance to continuous group, with the steerable filters (Weiler et al., 2018; Cohen et al., 2019), B-spline

interpolation (Bekkers, 2020) or Lie Group Theory (Bekkers, 2020; Finzi et al., 2020). A similar path to scaling equivariance is explored, though scaling is no longer intrinsically periodic. Deep scale space (Worrall & Welling, 2019) defined a semi-symmetry group to roughly obtain the scale equivariance, while Sosnovik et al. (2020) applied steerable CNNs on scaling. However, integrating equivariance to rotations and scalings at the same time results in a bigger group (e.g., a rotation group with M points and a scaling group with N points results in $M \times N$ points for the joint rotation and scaling group), making the task more difficult. Furthermore, certain "weight-sharing" techniques based on group convolution are computationally and memory storage intensive.

Despite the difficulties, empowering the model with equivariance of rotation and scaling together can be very beneficial. For example, in object detection, the distance changes between the camera and the object, or the random rotations of the object, can largely influence the method's accuracy.

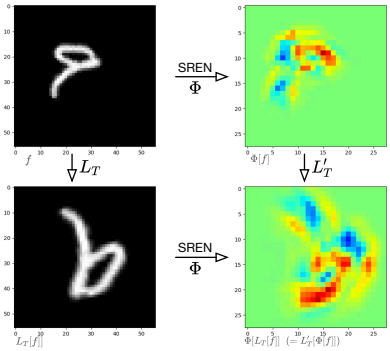

Figure 1: Sim(2) Equivariance property visualization: Our SREN method can inherently preserve the structure information of the input. Therefore, for the training of an object with a certain shape, our method can handle all the distorted(rotate, scaling, and translation) objects without further training.

In this paper, we aim to propose a CNNs that is continuously equivariant with rotation and scaling. Such work will fill a void in the area of equivariant. To achieve this, we provide a theory and analysis to guarantee that the network preserves the inherent equivariance property. Based on this, we propose a Scale and Rotation Equivariant Network (SREN) architecture. The method avoids the abovementioned limitations and does not sizably increase computational complexity. Specifically, We first designed a scalable Fourier-Argand representation. The expression of the basis makes it possible to operate the angle and scale in one shot. Based on this, we propose a new convolution-like operator that is slightly different while functionally similar to the traditional convolution. Furthermore, we show that the computational complexity is similar to convolution, so this method can easily replace the typical network structure. Finally, the method we designed can model rotation and scale scaling. Our model can achieve consistent results when tested with datasets with different transformations (such as rotation and scaling).

The main contributions in this paper are summarized as follows:

- We propose the scalable Fourier-Argand representation. This representation allows for achieving the similarity equivariance property.
- We propose the SimConv operator, together with the scalable Fourier-Argand filter, formed as the Scale Rotation Equivariant Network (SREN) architecture, which is an equivariant network for rotation and scaling.
- The method is very different from the group-convolutional neural networks. This provides a new possible path to solve the problem for the community and is not limited to the idea of group convolution.

2 Related Work

Group convolution Unlike the previous methods, which can achieve equivariance only in one aspect, our goal is to ensure rotation and scaling equivariant in a unique network. A possible direction is the application of group theory to achieve equivariance. Cohen & Welling (2016) introduced group convolution and enforced equivariance to small and discrete groups of transformation, i.e., rotations by multiples of 90 degrees. Subsequent works aim at generalizing the equivariantZhou et al. (2017) and also focus on continuous group coupled with the idea of steerable filters(Cohen & Welling, 2017). To achieve this purpose, Lie group theory, such as LieConv citepfinzi2020generalizing, is also presented. These studies allow for precise equivariance, but only for compact groups. In contrast to the rotating group, the scaling group is non-compact. And methods usually treat it as a semi-group and approximately achieve the truncated scaling equivariance. TridentNet (Li et al., 2019) gets scale invariance by sharing weights among kernels with different dilation rates. Scale-space theory

(Lindeberg, 2013) is also bring out by Worrall & Welling (2019) consider moving band-limits caused by rescaling in achieving semi-group equivariance. Bekkers (2020) generate the G-CNNs for arbitrary Lie groups by B-spline basis functions. With different settings, it can achieve scale or rotation equivariance. SiamSE(Sosnovik et al., 2021c) equip the Siamese network with additional built-in scale equivariance. Although Sosnovik et al. (2021b;a) replacing weight sharing scheme with dilated filters can parallel the process, and thus O(1) in terms of time, the overall computational load is still increased concerning the group size. Beyond all these methods, expanding the group to a larger space can increase the overhead computation.

Steerable filters Works with steerable filters are also used to achieve equivariance. The underlying insight is to use different linear combinations of fixed basis to represent filters of different sizes or rotation angles. H-Net (Worrall et al., 2017) uses complex circular harmonics while SESN (Sosnovik et al., 2020) uses the Hermite polynomials as filter bases to achieve equivariance. This is a bit related to our method. However, due to the lack of a proper existing basis in the image processing field, this path to achieving rotation-scaling equivariant is also difficult. Other works improved the equivariant on different aspects. Polar transformer networks (Esteves et al., 2018) generalizes group-equivariance to rotation and dilation. Attentive group convolutions (Romero et al., 2020) Use the attention mechanism to generalize the group convolution. Some other techniques, for example, Shen et al. (2020) propose using the partial differential operator to maintain the equivariance. Jenner & Weiler (2022) use the partial differential operators (PDOs) to design the equivariant CNNs effectively. Gao et al. (2022) presents a roto-scale-translation equivariant CNN, but it expands filters of the G-CNN in the scale dimension with a truncated interval.

Differences between related works and ours. Unlike the previous methods our goal is to ensure continuous rotation and scaling equivariant in a unique network. However, methods that are based on modifying the filters used in the neural network to achieve some form of equivariance, require more parameters to learn and are computationally more expensive. Instead, we propose using a scalable and steerable filter representation (scalable Fourier-Argand) to modify the convolution operator (SimConv) so that it embodies scale, rotation, and shift equivariance. This transformation does not introduce new learnable parameters. Thus, we can then achieve the rotation and scale equivariance efficiently that none of the other methods enjoy.

3 PRELIMINARIES AND NOTATION

This section clarifies the notations about the sim(2) transformation and the convolution that is often mentioned later. The property of equivariant and invariant are also formulated explicitly, which is derived from our proposed method.

3.1 SIM(2) TRANSFORMATION

In Euclidean geometry, two objects can be transformed into each other by the similarity transformation if they share the same shape. This similarity is critical in instance-level computer vision, as objects do not change with scale, translation, and rotation transformations. This motivates us to study the similarity equivariant. Let's consider the Sim(2) group that the invertible translation matrix is as follows,

$$\mathbf{T} = s \begin{bmatrix} \mathbf{R}^\top & -s^{-1}\mathbf{R}^\top\mathbf{t} \\ \mathbf{0} & s^{-1} \end{bmatrix}, \text{and} \quad \mathbf{T}^{-1} = s^{-1} \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & s \end{bmatrix} \in \mathrm{Sim}(2) \subset \mathbb{R}^{3\times 3} \tag{1}$$

where $\mathbf{R} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \in \mathbb{R}^{2\times 2}$ denotes the rotation matrix, $\mathbf{t} \in \mathbb{R}^2$ is a translation vector and $s \in \mathbb{R}$ is the scale factor. In this way, rotation (by θ angles) , scaling (by s times) and translation (by t pixels) are integrated in one matrix and can written as the matrix multiplication of three shape-preserving matrix, i.e., $\mathbf{T} = \mathbf{A}_s\mathbf{Y}_\mathbf{t}\mathbf{R}_\theta$. Now let's consider a spatial index on 2D image plane $\tilde{x} = [x_1, x_2]^\top \in \mathbb{R}^2$, we extend its domain to triples as $\mathbf{x} = [\tilde{x}, 1]^\top \in \mathbb{R}^3$. Then for an input signal f, we define a linear transformation $L_\mathbf{T} : \mathbb{L}_2(X) \to \mathbb{L}_2(X)$ that transform feature maps $f \in \mathbb{L}_2(X)$ on some space X, written as,

$$L_\mathbf{T}[f](\mathbf{x}) = f(\mathbf{T}^{-1}\mathbf{x}) \tag{2}$$

The left-hand side can be regarded as a linear transformation $L_\mathbf{T}$ acting on a set of feature maps f, and the right-hand side can be interpreted as finding the value of the feature map f at the point

$\mathbf{T^{-1}x}$. Additionally, we have $L_{\mathbf{T}}[\cdot] = L_{(\mathbf{A}_s\mathbf{Y_t}\mathbf{R}_\theta)}[\cdot] = L_{\mathbf{A}_s}[L_{\mathbf{Y_t}}[L_{\mathbf{R}_\theta}[\cdot]]]$. When considering the transformation we have,

$$f(\mathbf{T^{-1}x}) = f\left([s^{-1}(\mathbf{R}\tilde{x} + 1\mathbf{t})^\top, 1]^\top\right) \tag{3}$$

This corresponds to a rotation transformation followed by a translation and scaling.

3.2 Formulation of Convolution

Let's consider a stack of two-dimensional features as a function $f(\cdot) = \{f_c(\cdot)\}_{c=1}^N : \mathbb{R}^2 \times 1 \to \mathbb{R}^N$, where N is the number of the channel. Similarly, a filter bank contains M elements in convolutional layer can be formalized as $\varphi = \{\psi_k\}_{k=1}^M$, where each filter with N channels can be written as $\psi_k = \{\psi_{k,c}\}_{c=1}^N$, and $\psi_k : \mathbb{R}^2 \times 1 \to \mathbb{R}^N$ is a vector valued output filter. Then, for a continuous input with N channels, we regard the spatial cross-correlation between the input and the continuous filter bank Ψ as an operator $\Phi[\cdot] = [\cdot \star \varphi] : \mathbb{R}^N \to \mathbb{R}^M$, written as follow,

$$\Phi[f](\mathbf{x}) = [f \star \varphi](\mathbf{x}) = \{\sum_{c=1}^{N} \int_R f_c(\mathbf{x} - \mathbf{t})\psi_{c,k}(\mathbf{t})\mathrm{d}\mathbf{t}\}_{k=1}^M \tag{4}$$

Without loss of generality, we can set $N = M = 1$ and simplify the above formula with the following formula,

$$\Phi[f](\mathbf{x}) = [f \star \varphi](\mathbf{x}) = \int_R f(\mathbf{x} - \mathbf{t})\psi(\mathbf{t})\mathrm{d}\mathbf{t} \tag{5}$$

To clarify, although $\mathbf{t}$ is three dimension vector, this integral is still a double integral (along the first and second dimensions), the same as the regular convolution. Since the last dimension of $\mathbf{t}$ is only a placeholder. We use this simplified formula in the remaining paper to make the deduction more readable.

3.3 Equivariance and Invariance Property

Definition 3.1 (Equivariance)**.** *Given an operator $\Phi : \mathbb{L}_2(N) \to \mathbb{L}_2(M)$, the operator Φ is equivariant to the transform $L_{\mathbf{T}}$ if for any $\mathbf{x} \in \mathbb{R}^{d+1}$, we can find a predictable transform $\tilde{L}_{\mathbf{T}}$, such that the equation below holds.*

$$\Phi[L_{\mathbf{T}}[f]](\mathbf{x}) = \tilde{L}_{\mathbf{T}}[\Phi[f]](\mathbf{x}) \tag{6}$$

If $L_{\mathbf{T}} = \tilde{L}_{\mathbf{T}}$, we can also say that these two operators are commutable. This equivariance property provides structure-preserving properties for the network.

Definition 3.2 (Invariance)**.** *Given an operator $\Phi : \mathbb{L}_2(N) \to \mathbb{L}_2(M)$, the operator Φ is invariant to the transform $L_{\mathbf{T}}$ if for any $\mathbf{x} \in \mathbb{R}^{d+1}$, the equation below holds.*

$$\Phi[L_{\mathbf{T}}[f]](\mathbf{x}) = \Phi[f](\mathbf{x}) \tag{7}$$

This paper aims to design a convolution-like operation that is similar in function to convolution without lifting the intermediate variable size of the network (This is common in group methods.), as well as satisfies the equivariance or invariance property for any similar transform.

4 Method

This section discussed the possibility of making the traditional convolutional neural network to be equivariance in rotation, scale, and translation simultaneously. The underlying intuition behind the method is: we first obtain the local scale and orientation of the image. Then, this local information is used to adapt the scale and direction of the filter used for the convolution. Moreover, this (image-dependent) spatially-varying convolution has an efficient implementation.

4.1 Scalable Fourier-Argand Representation

We first proposed a representation that we called the scalable Fourier-Argand Representation. The insight of designing this exact and steerable representation is that we can use this representation to retrieve the local geometric information and use it as a covariance indicator in Equation (12) for further usage. We first define scalable Fourier-Argand Representation as follows,

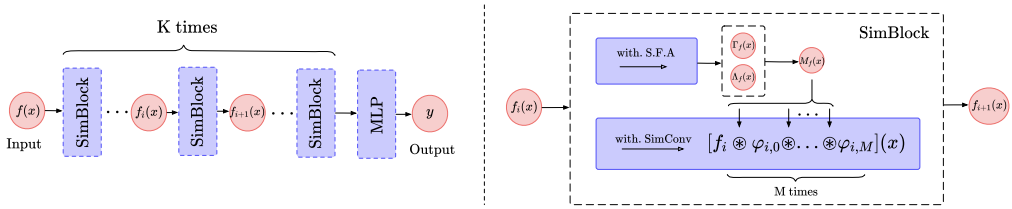

Figure 2: An overview of SREN: The architecture consists of multiple SimBlocks. In each block, we use the proposed scalable Fourier-Argand filter (see Section 4.1) to extract the geometry information $\mathbf{M}_f(\mathbf{x})$. We then combine this indicator with our similarity convolution (see Section 4.2) structure to make the network similarity equivariant. We add a head layer lastly to convert equivariant output to the invariant output (see Section 4.3).

Definition 4.1 (The scalable Fourier-Argand representation)**.** *Consider a square-integrable function h, and express it as $h(r, \theta)$ in polar coordinates. We propose its scalable Fourier Argand representation as the following series form,*

$$h(r,\theta) = \sum_{k_1,k_2\in\mathbb{Z}} \left(h_{k_1,k_2} r^{m_{k_1}} \exp\left(i(k_1\theta + k_2 \frac{2\pi \ln r}{\ln b/a}) \right) \right) = \sum_{k_1,k_2\in\mathbb{Z}} H_{k_1,k_2}(r,\theta) \tag{8}$$

Where $\mathbb{Z}$ denotes the set of all integers. One has flexibility in choosing any m_k as a constant value. The function has limited support that $(r, \theta) \in [0, 2\pi[\times[a, b[$. In reality, k_1, k_2 can be truncated for a subset of integers and approximate h.

Proposition 4.1. *Let $h_{k_1,k_2} : \mathbb{Z}^2 \to \mathbb{C}$ as the coefficient of each item, and we can compute it as follows,*

$$h_{k_1,k_2} = \frac{1}{\ln \frac{b}{a}} \int_{\ln a}^{\ln b} \frac{1}{2\pi} \int_0^{2\pi} h(e^\rho, \theta) \exp\left(-ik_1\theta - ik_2 \frac{2\pi\rho}{\ln b/a} - \rho m \right) d\theta d\rho \tag{9}$$

The proof of this can be found in Appendix A. This formula can represent the filter as a composition of elementary feature basis. The part about θ can be seen as the Fourier series of filters on the Argand plane, which is inspired by Zhao & Blu (2020). The filter itself is in relation to have a connection with the harmonic filterWorrall et al. (2017). For the scaling part, We use the logarithmic method to make the scaling of the size a linear shift. We also restrict the support of the function as a plane that radius ranges in $[a, b]$, where $a \to 0^+$. In practice, we set $m_k = -1$ that acts like a window function, which ensures that the function can vanish at infinity. The above explains why we call it the scalable Fourier-Argand representation.

One benefit of the expression we propose is that its basis functions are steerable for rotation and scalable for scaling. Specifically, consider the transformation matrix $\mathbf{T} = \mathbf{A}_s \mathbf{Y_0} \mathbf{R}_\alpha$ that contains only rotation and scaling, we have,

$$\begin{aligned} L_{\mathbf{T}}[H_{k_1,k_2}](r,\theta) &= h_{k_1,k_2} \cdot \exp(ik_1(\theta-\alpha) + ik_2 \frac{2\pi \ln r/s}{\ln b/a}) \cdot (rs^{-1})^{m_{k_1}} \\ &= H_{k_1,k_2}(r,\theta) \cdot \exp(-ik_1\alpha - ik_2 \frac{2\pi \ln s}{\ln b/a}) s^{-m_{k_1}} \end{aligned} \tag{10}$$

This equation holds for any $\alpha \in [0, 2\pi)$ and $s \in (a, b)$. We can easily verify the following proposition,

Proposition 4.2. *For a continuous filter $h \in \mathbb{R}^2$ that can be decomposed by a set of basis, let $\mathbf{T} = \mathbf{A}_s \mathbf{Y_0} \mathbf{R}_\alpha$ as a transformation matrix. Then the transformed filter of h, noted as $L_{\mathbf{T}}[h]$, can still be represented by the same basis, with a steerable linear combination:*

$$\begin{aligned} L_{\mathbf{T}}[h](r,\theta) &= \sum_{k_1,k_2} L_{\mathbf{T}}[H_{k_1,k_2}](r,\theta) \\ &= \sum_{k_1,k_2} H_{k_1,k_2}(r,\theta) \cdot \left(\exp(-ik_1\alpha - ik_2 \frac{2\pi \ln s}{\ln b/a}) s^{-m_{k_1}} \right) \end{aligned} \tag{11}$$

With this property, we can fix the basis H_{k_1,k_2} and estimate the filter h for rotation and scaling by using different linear combinations. Furthermore, if we want to convolve the input signal f with the filter h and its various $L_T[h]$, we can pre-convolve the image with the basis of the original filter H_{k_1,k_2}. Besides, the normalized cross-correlation is a more robust substitution of traditional convolution, denoted as $\star$. This allows us to obtain the intermediate variable f_{k_1,k_2} by the following formula,

$$f_{k_1,k_2}(\mathbf{x}) = [f \star H_{k_1,k_2}](\mathbf{x}) = \frac{[f * H_{k_1,k_2}](\mathbf{x}) - \mu_i(\mathbf{x})\mu_{H_{k_1,k_2}}}{\sigma_i(\mathbf{x})\sigma_{H_{k_1,k_2}}} \tag{12}$$

Here $*$ is the traditional convolution, μ, and σ are the mean and variance of the basis filters or signal. With these signal basis, we can then obtain the optimal orientation and scale by calculating the argmax of the combination of basis, written as follows,

$$[\Lambda_f(\mathbf{x}), \Gamma_f(\mathbf{x})] = \arg\max_{(\lambda,\gamma)} \sum_{k_1,k_2=-K}^{K} f_{k_1,k_2}(\mathbf{x}) \cdot c_{k_1,k_2}(\lambda,\gamma) = \arg\max_{\gamma,\lambda} \mathbf{c}_{\gamma,\lambda}\mathbf{F} \tag{13}$$

Where $c_{k_1,k_2}(\lambda,\gamma) = \exp(-ik_1\gamma - ik_2\frac{2\pi \ln\lambda}{\ln b/a})\lambda^{-m}$ is a coefficient that only relys on λ and γ. $\mathbf{c}_{\gamma,\lambda}$ and $\mathbf{F}$ are vectors of all possible f_{k_1,k_2} and c_{k_1,k_2} $\Lambda_f(x), \Gamma_f(x)$ can be understood as the projection of signal f for the orientation and scale aspect. These two indicators meet the following properties,

Lemma 4.1. *Let $\mathbf{T} = \mathbf{A}_s\mathbf{Y}_t\mathbf{R}_\alpha$ as a similarity transformation. Then for a input image f and its distorted version $L_T[f](x)$, for any position x, we can have a relationship of this pair of images by following property,*

$$\begin{aligned} \Lambda_{L_\mathbf{T}[f]}(\mathbf{x}) &= \Lambda_f(\mathbf{T^{-1}x}) \cdot s \\ \Gamma_{L_\mathbf{T}[f]}(\mathbf{x}) &= \Gamma_f(\mathbf{T^{-1}x}) + \alpha \end{aligned} \tag{14}$$

The proof can be found in Appendix B. This is the condition that we achieved and applied later in Section 4.2.

4.2 Similarity Convolution

We propose the similarity convolution (SimConv) as an alternative to traditional convolution. We first list the feature of SimConv that we expect as follows: 1). Theoretically, it should have equivariance properties of rotation, scale, and translation. 2). It should be a convolution-like operation, containing some learnable parameters and extracting image features by "blending" one function with another. 3). Its computational complexity should be close to traditional convolution. Therefore, unlike group convolution, it does not face the problem of computational disaster when extending the group size. These criteria drive us design the similarity convolution as follows,

Definition 4.2 (Similarity convolution)**.** *The similarity convolution between the input signal f and the filter φ is defined as*

$$[f \circledast \varphi](\mathbf{x}) := \int_R f(\mathbf{x} + \mathbf{M}_f(\mathbf{x})\mathbf{t})\varphi(\mathbf{t})\mathrm{d}\mathbf{t} = \Phi[f](\mathbf{x}) \tag{15}$$

where $\mathbf{M}_f(\mathbf{x})$ is defined as following pixel-wise matrix,

$$\mathbf{M}_f(\mathbf{x}) = \mathbf{A}_{(\Lambda_f(\mathbf{x}))}\mathbf{R}_{(\Gamma_f(\mathbf{x}))} \in \mathrm{Sim}(2) \tag{16}$$

This SimConv has a convolution-like structure between feature f and the learnable filter φ. This is clear when we do variable substitude with $\mathbf{t} = \mathbf{M}_f^{-1}(\mathbf{x})\tilde{\mathbf{t}}$. And it can degenerate to traditional convolution when setting $\mathbf{M}_f(\mathbf{x})$ to Identity matrix for all x. We note this SimConv as an operator $\Phi[\cdot] = [\cdot \circledast \varphi] : \mathbb{R}^N \to \mathbb{R}^M$, where we set $M = N = 1$ for easier read. This newly defined convolution is, in fact, equivariant for rotation, scaling, and translation. We verify this property below.

Considering $\mathbf{M}_f(\mathbf{x})$ in Equation (16) and combined with the condition of Equation (14), we can easily prove that

$$\begin{aligned} \mathbf{M}_{L_\mathbf{T}[f]}(\mathbf{x})\mathbf{T^{-1}} &= \mathbf{A}_{(\Lambda_{L_\mathbf{T}[f]}(\mathbf{x}))}\mathbf{R}_{(\Gamma_{L_\mathbf{T}[f]}(\mathbf{x}))}\mathbf{T^{-1}} \\ &= \mathbf{A}_{(\Lambda_f(\mathbf{T^{-1}x}))}\mathbf{A}_s\mathbf{R}_{(\Gamma_f(\mathbf{T^{-1}x}))}\mathbf{R}_\alpha\mathbf{T^{-1}} \\ &= \mathbf{A}_{(\Lambda_f(\mathbf{T^{-1}x}))}\mathbf{R}_{(\Gamma_f(\mathbf{T^{-1}x}))} = \mathbf{M}_f(\mathbf{T^{-1}x}) \end{aligned} \tag{17}$$

This leads to the following summarized condition: $\mathbf{T^{-1}} = \mathbf{M}^{-1}_{L_{\mathbf{T}}[f]}(\mathbf{x}) \cdot \mathbf{M}_f(\mathbf{T^{-1}x})$,$\forall \mathbf{x} \in \mathbb{R}^2 \times 1$. If we apply a similarity transformation $L_{\mathbf{T}}$ that rotates the signal by α degree centered at t, as well as scaling by s, to the signal f following Equation (2), then for signals after convolution, we have

$$\begin{aligned}\Phi[L_{\mathbf{T}}[f]](\mathbf{x}) = [L_{\mathbf{T}}[f] \circledast \varphi](\mathbf{x}) &= \int_R L_{\mathbf{T}}[f](\mathbf{x} + \mathbf{M}_{L_{\mathbf{T}}[f]}(\mathbf{x})\mathbf{t})\varphi(\mathbf{t})\mathrm{d}\mathbf{t} \\ &= \int_R f(\mathbf{T}^{-1}(\mathbf{x} + \mathbf{M}_{L_{\mathbf{T}}[f]}(\mathbf{x})\mathbf{t}))\varphi(\mathbf{t})\mathrm{d}\mathbf{t} = \int_R f(\mathbf{T}^{-1}\mathbf{x} + \mathbf{T}^{-1}\mathbf{M}_{L_{\mathbf{T}}[f]}(\mathbf{x})\mathbf{t})\varphi(\mathbf{t})\mathrm{d}\mathbf{t}\end{aligned} \tag{18}$$

Besides, the commutator operator of the above can similarly deduct as,

$$\begin{aligned}L_{\mathbf{T}}[\Phi[f]](\mathbf{x}) = L_{\mathbf{T}}[[f \circledast \varphi]](\mathbf{x}) &= L_{\mathbf{T}}[\int_R f(\mathbf{x} + \mathbf{M}_f(\mathbf{x})\mathbf{t})\varphi(\mathbf{t})\mathrm{d}\mathbf{t}] \\ &= \int_R f(\mathbf{T^{-1}x} + \mathbf{M}_f(\mathbf{T^{-1}x})\mathbf{t})\varphi(\mathbf{t})\mathrm{d}\mathbf{t} = \int_R f(\mathbf{T^{-1}x} + \mathbf{T}^{-1}\mathbf{M}_{L_{\mathbf{T}}[f]}(\mathbf{x})\mathbf{t})\varphi(\mathbf{t})\mathrm{d}\mathbf{t}\end{aligned} \tag{19}$$

Replace the second $\mathbf{T}$ in Equation (19), and compare with Equation (18), we can conclude that

$$\Phi[L_{\mathbf{T}}[f]](\mathbf{x}) = L_{\mathbf{T}}[\Phi[f]](\mathbf{x}) \tag{20}$$

This shows that for the input signal f, if we apply the similarity transform and then apply the SimConv, it is equivalent to first applying the SimConv and then the transform. So with scalable Fourier Argand representation, the proposed Similarity Convolution satisfies the equivalence property. Moreover, if each layer in the network satisfies this property, then the transformation can be passed from the first layer to the last layer, which can be expressed as the following formula,

$$\begin{aligned}f_n(\mathbf{x}) &= [L_{\mathbf{T}}[f_0] \circledast \varphi_0 \circledast ... \circledast \varphi_n](\mathbf{x}) \\ &= [L_{\mathbf{T}}[f_0 \circledast \varphi_0 \circledast ... \circledast \varphi_n]](\mathbf{x})\end{aligned} \tag{21}$$

Moreover, there are few choices to convert the equivariant features into the invariant features, formulated as $P \circ L_{\mathbf{T}} = P$. One option is to add an adaptive max pooling layer at the end of the network. Let $P[\cdot]$ = torch.nn.AdaptiveMaxPool2d(1) as a function that takes the maximum response over the entire spatial domain. Since the maximum value won't be affected by the position distortion of the feature, the output is invariant. And from Equation (20) we have,

$$P[\Phi[L_{\mathbf{T}}[f]]](\mathbf{x}) = P[L_{\mathbf{T}}[\Phi[f]]](\mathbf{x}) = P[\Phi[f]](\mathbf{x}) \tag{22}$$

This means the transformation matrix $\mathbf{T}$ will not influence the output, which can be beneficial in tasks such as classification.

4.3 Discretization method

Although the continuous formulation in the previous section is necessary to go, from the intuitive approach (find scale + orientation, then filter accordingly), to the efficiently implementable formulation (Equation (14)) through a change of variables in an integral, the digital images or feature maps are usually discrete data aligned on the mesh grid. Therefore, we detailed the discretization of the integral and approximation implementation. We rewrite Equation (15) in discrete form as follows,

$$\Phi[f](x) = [f \circledast \varphi](\mathbf{x}) = \frac{V}{n} \sum_{t \in \mathcal{R}} f(\mathbf{x} + \mathbf{M}_f(\mathbf{x})\mathbf{t})\varphi(\mathbf{t}) \tag{23}$$

Where $\mathcal{R}$ is the support of φ. Without loss of generality, take 3×3 convolution as an example, then $\mathcal{R} = \{(t_x, t_y, 1)^T | t_x, t_y \in \{-1, 0, 1\}\}$. n is the number of the elements of the set $\mathcal{R}$. Let $\mathbf{y_t} = \mathbf{x} + \mathbf{M}_f(\mathbf{x})t$. Generally, this is a fractional location index. So estimating this value and its gradient becomes a problem. Here we approximate $f(\mathbf{y_t})$ using the bilinear interpolation technique, written as, $f(\mathbf{y_t}) = \sum_{\mathbf{m}} G(\mathbf{y_t}, \mathbf{m}) f(\mathbf{m})$Where $\mathbf{m} = (m_1, m_2, 1)^T \in \mathbb{Z}^3$. G is bilinear interpolation kernel and its formula is $G(m, n) = g(m_1, n_1)g(m_2, n_2)$, where $g(a, b) = max(0, 1 - |a - b|)$. So the similarity convolution becomes

$$\Phi[f](\mathbf{x}) = [f \circledast \varphi](\mathbf{x}) = \frac{V}{n} \sum_{\mathbf{t} \in \mathcal{R}} \sum_{\mathbf{m}} G(\mathbf{y_t}, \mathbf{m}) f(\mathbf{m})\varphi(\mathbf{t}) \tag{24}$$

With this implementation, we also make the similarity convolution differentiable. The gradient of the input is written as,

$$\frac{\partial \Phi[f](\mathbf{x})}{\partial \mathbf{x}} = \frac{V}{n} \sum_{\mathbf{t} \in \mathcal{R}} \sum_{\mathbf{m}} \frac{\partial G(\mathbf{y_t}, \mathbf{m})}{\partial \mathbf{x}} f(\mathbf{m}) \varphi(\mathbf{t}) \tag{25}$$

Here G is a differentiable function, and since it is non-zero only when m aligns on the grid of y_t, it does not require too much computation.

5 EXPERIMENTS

5.1 CHARACTER RECOGNITION TASK

Dataset. MNIST-ROT-12K DATASET (Larochelle et al., 2007) is usually used to validate the rotation equivariant algorithms. Yet this dataset is insufficient to verify equivariant on both scaling and rotation. Thus, to better facilitate model comparison, we modify the original MNIST dataset following a similar path and construct the SRT-MNIST DATASET. Specifically, we first zero-padded the original dataset to 56×56 pixel size. Then we keep training set still and randomly rotate, scaling, and translate each image by a range of $\theta = [0, 2\pi), s = [1, 2[$ and $t = \pm 10$ respectively. This out-of-distribution setting of the test image can sufficiently evaluate the model's generalization ability.

Experiments setup. We use a ResNet-18 He et al. (2016) as the architecture. For every convolution layer, we replace it with our SimConv while maintaining the same trainable parameters. All layers share the same scalable Fourier-Argand filters. We use Adam optimizer (Kingma & Ba, 2015) with a weight decay of 0.01. Weights are initialized by Xavier (Glorot & Bengio, 2010). The learning rate is set to 0.01 and is decayed by a factor of 0.1 every 50 epochs. We set the batch size as 128, and stopped the training after 200 epochs. All model parameters are 11.68M, and the FLOPs is 0.12G for an input image with 56×56.

Table 1: Generalization ability test on SRT-MNIST

Methods	Type of the test set.			
	MNIST	R-MNIST	S-MNIST	SRT-MNIST
CNNs	99.46	44.41	73.21	33.56
SO(2)-Conv	99.23	**97.18**	72.85	70.72
$\mathbb{R}^*$-Conv	99.31	35.23	**99.21**	39.2
SREN	99.12	96.91	98.48	**92.3**
SREN+	99.42	98.3	99.28	95.1

Generalization ability study. We conduct the ablation study to verify our method's equivariance property on rotation and scaling independently and concurrently. In Table 1, R-MNIST, S-MNIST means we only apply rotation or scaling to the test dataset. SO(2)-Conv and $\mathbb{R}^*$-Conv methods have the same structure as our proposed SREN but set all the Γ_f or λ_f to unitary, which is a convenient way to disable the equivariance property partially. SREN+ indicates our method with a data augmentation technique randomly rotates for ± 30 degrees, scaling for $[0.8, 1.2]$ times. We can see that SO(2)-Conv and $\mathbb{R}^*$-Conv achieved equivariance property on rotation or scaling. With the equivariant mechanism, our SREN algorithm can reach an accuracy rate of over 95% on every dataset, whereas the CNNs can only overfit the original dataset with limited generalization ability.

Equivariance error analysis. We numerically validate the quality of equivariance by the equivariant error to reveal how stable the equivariance property of the method is and the main factor that affects the stability. We define the equivariant error by measuring the normalized $L-2$ distance as follows,

$$\text{Error} = \frac{||L_{\mathbf{T}}[\Phi[f]] - \Phi[L_{\mathbf{T}}[f]]||_F^2}{||L_{\mathbf{T}}[\Phi[f]]||_F^2} \tag{26}$$

where $||\cdot||_F$ is the Frobenius norm. The formula is the relative percentage error of the two obtained features after the input is first convolved, then transformed, and after the input is first distorted and then convolved. We compare the k-th layer feature with the convolutional network. The result is shown in Figure 3. It can be seen that the average error is lower than 0.01. This shows that our network can be equivariant in good quality. Besides, we test a specific case that image is rotated by 90 degrees. In this case, there will not be a discretization problem for rotation, and the equivariance error can be as small as 1.73×10^{-6} and almost negligible. This shows that our method can be extremely accurate for equivariant without discretization.

5.2 NATURAL IMAGE CLASSIFICATION

Experiments setup. To evaluate the generalization ability of our method, we do experiments on the challenge of the STL-10 dataset Coates et al. (2011). The labeled subset is an excellent choice to evaluate how efficient the network can use these limited training samples and how much the network's generalization ability is when the training set is small. We evaluate methods by in-distribution testing (ID) that unchange the test dataset, and out-of-distribution testing that randomly rotates and scales the dataset. The OOD test measures the ability of a method to handle the never-seen inputs.

Table 2: The comparison on STL-10 dataset: Our approach is achieved coutinuous equivariant on join rotation and scaling.

Methods	$\mathcal{R}$-Equi	S-Equi	Conti	ID Accuracy (%)	OOD Accuracy (%)
ResNet-16	✗	✗	✗	82.66 ± 0.53	37.63 ± 1.95
RDCF	✓	✗	✗	83.66 ± 0.57	51.12 ± 4.21
SESN	✗	✓	✓	83.79 ± 0.24	47.26 ± 0.63
SDCF	✗	✓	✗	83.83 ± 0.41	43.60 ± 0.87
RST-CNN	✓	✓	✗	84.08 ± 0.11	58.31 ± 3.62
SREN	✓	✓	✓	$\mathbf{85.25 \pm 0.61}$	$\mathbf{63.42 \pm 2.57}$

We use the ResNet (He et al., 2016) with 16 layers as our backbone, and use our SimConv layers to replace all convolutional layers. We trained the network for 1000 epochs with a batch size of 128. And we use Adam as the optimizer. The initial learning rate is set to 0.1 and adjusted with a cosine annealing schedule during training. Following Zhu et al. (2019), data augmentation without scaling and rotation is also applied. We compare our method with Rotation Decomposed Convolutional Filters network (RDCF) (Cheng et al., 2019), Scale-Equivariant Steerable Networks (SESN) (Sosnovik et al., 2020), Scale Decomposed Convolutional Filters network (SDCF) (Zhu et al., 2019), Roto-Scale-Translation Equivariant CNNs (Gao et al., 2022). All methods' backbone is set to ResNet (He et al., 2016) with 16 layers to make the model parameters comparable and make the comparison fair enough.

Results & Discussion. Table 2 demonstrates our main results compared to recent baselines. $\mathcal{R}$-Equi and S-Equi indicate whether method achieves equivariance property in rotation or scaling. *Conti* means whether the method achieves equivariance at a continuous scale or rotation. As we can see, our method achieved the highest accuracy among all other approaches, especially for the out-of-distribution test. This shows that our method has a good generalization ability. Besides, to be noticed that we make a comparison with methods that "partially" achieve equivariant since there are no like-for-like methods (achieving *rotation* and *rotation* scaling equivariant in the *continuous* group) to compare. MacDonald et al. (2022) is another paper that guarantee the equivariance to any finite-dimensional Lie group, but the memory efficiency limits it to scale up to a large network and make an affair comparation. It is not easy to achieve this property efficiently, and this paper is one of the few ways to achieve this property. Being able to achieve this property itself is already another state-of-the-art.

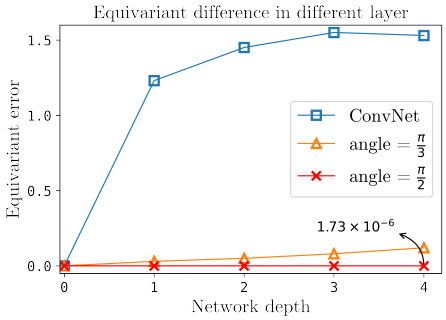

Figure 3: Equivariance error: Our method shows good equivariant quality with a multi-layer network. The error can reach the 10^{-6} level if there is no discretization approximation.

6 CONCLUSION

Although there have been numerous studies on how to achieve rotation and scale equivariant, achieving continuous equivariant in rotation and scaling is novel, to our knowledge. In this paper, we propose to develop scalable steerable filters based on the Fourier-Argand representation and to use the local scale and orientation provided by these filters to empower the convolution operator with local scale and rotation equivariant: SimConv. Mathematical and experimental analyses are detailed to explain why it works and to what extent it can achieve the desired property.

REFERENCES

Erik J Bekkers. B-spline cnns on lie groups. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=H1gBhkBFDH.

Xiuyuan Cheng, Qiang Qiu, Robert Calderbank, and Guillermo Sapiro. Rotdcf: Decomposition of convolutional filters for rotation-equivariant deep networks. In *International Conference on Learning Representations 2019 (ICLR'19)*, 2019.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 215–223. JMLR Workshop and Conference Proceedings, 2011.

Taco Cohen and Max Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pp. 2990–2999. PMLR, 2016.

Taco S. Cohen and Max Welling. Steerable CNNs. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=rJQKYt5ll.

Taco S Cohen, Mario Geiger, and Maurice Weiler. A general theory of equivariant cnns on homogeneous spaces. *Advances in neural information processing systems*, 32, 2019.

Carlos Esteves, Christine Allen-Blanchette, Xiaowei Zhou, and Kostas Daniilidis. Polar transformer networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=HktRlUlAZ.

Marc Finzi, Samuel Stanton, Pavel Izmailov, and Andrew Gordon Wilson. Generalizing convolutional neural networks for equivariance to lie groups on arbitrary continuous data. In *International Conference on Machine Learning*, pp. 3165–3176. PMLR, 2020.

Liyao Gao, Guang Lin, and Wei Zhu. Deformation robust roto-scale-translation equivariant CNNs. *Transactions on Machine Learning Research*, 2022. URL https://openreview.net/forum?id=yVkpxs77cD.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256. JMLR Workshop and Conference Proceedings, 2010.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Erik Jenner and Maurice Weiler. Steerable partial differential operators for equivariant neural networks. In *ICLR*, 2022.

Angjoo Kanazawa, Abhishek Sharma, and David Jacobs. Locally scale-invariant convolutional neural networks. *arXiv preprint arXiv:1412.5104*, 2014.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

Hugo Larochelle, Dumitru Erhan, Aaron Courville, James Bergstra, and Yoshua Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *Proceedings of the 24th international conference on Machine learning*, pp. 473–480, 2007.

Yanghao Li, Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Scale-aware trident networks for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6054–6063, 2019.

Tony Lindeberg. *Scale-space theory in computer vision*, volume 256. Springer Science & Business Media, 2013.

Lachlan E MacDonald, Sameera Ramasinghe, and Simon Lucey. Enabling equivariance for arbitrary lie groups. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8183–8192, 2022.

David Romero, Erik Bekkers, Jakub Tomczak, and Mark Hoogendoorn. Attentive group equivariant convolutional networks. In *International Conference on Machine Learning*, pp. 8188–8199. PMLR, 2020.

Zhengyang Shen, Lingshen He, Zhouchen Lin, and Jinwen Ma. Pdo-econvs: Partial differential operator based equivariant convolutions. In *International Conference on Machine Learning*, pp. 8697–8706. PMLR, 2020.

Ivan Sosnovik, Michał Szmaja, and Arnold Smeulders. Scale-equivariant steerable networks. In *International Conference on Learning Representations*, 2020.

Ivan Sosnovik, Artem Moskalev, and Arnold Smeulders. Disco: accurate discrete scale convolutions. *arXiv preprint arXiv:2106.02733*, 2021a.

Ivan Sosnovik, Artem Moskalev, and Arnold Smeulders. How to transform kernels for scale-convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1092–1097, 2021b.

Ivan Sosnovik, Artem Moskalev, and Arnold WM Smeulders. Scale equivariance improves siamese tracking. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2765–2774, 2021c.

Maurice Weiler, Fred A Hamprecht, and Martin Storath. Learning steerable filters for rotation equivariant cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 849–858, 2018.

Daniel E Worrall and Max Welling. Deep scale-spaces: equivariance over scale. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 7366–7378, 2019.

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*, pp. 5028–5037, 2017.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pp. 818–833. Springer, 2014.

Tianle Zhao and Thierry Blu. The fourier-argand representation: An optimal basis of steerable patterns. *IEEE Transactions on Image Processing*, 29:6357–6371, 2020.

Yanzhao Zhou, Qixiang Ye, Qiang Qiu, and Jianbin Jiao. Oriented response networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 519–528, 2017.

Wei Zhu, Qiang Qiu, Robert Calderbank, Guillermo Sapiro, and Xiuyuan Cheng. Scaling-translation-equivariant networks with decomposed convolutional filters. *arXiv preprint arXiv:1909.11193*, 2019.

A PROOF OF SCALABLE FOURIER ARGAND REPRESENTATION

Definition A.1. *Consider a square integrable function h, and express it as $h(r,\theta)$ in polar coordinates. We propose its Scalable Fourier Argand representation as the following series form,*

$$h(r,\theta) = \sum_{k_1,k_2\in\mathbb{Z}} \left[h_{k_1,k_2} r^{m_{k_1}} \exp(i(k_1\theta + k_2 \frac{2\pi \ln r}{\ln b - \ln a})) \right] = \sum_{k_1,k_2\in\mathbb{Z}} H_{k_1,k_2}(r,\theta) \tag{27}$$

Where $h_{k_1,k_2} : \mathbb{Z}^2 \to \mathbb{C}$ is the coefficient of each item, which can be computed as follows,

$$h_{k_1,k_2} = \frac{1}{\ln b/a} \int_{\ln a}^{\ln b} \frac{1}{2\pi} \int_0^{2\pi} h(e^\rho,\theta) \exp(-ik_1\theta - ik_2 \frac{2\pi\rho}{\ln b/a} - \rho m) d\theta d\rho \tag{28}$$

Proof.

We start from Equation (27) and replace the polar coordinate r by an exponential term $r = e^\rho$, then $\rho = \ln r$ and we have,

$$h(e^\rho,\theta) = \sum_{k_1,k_2\in\mathbb{Z}} \left[h_{k_1,k_2} \exp(\rho m_{k_1}) \exp(i(k_1\theta + k_2 \frac{2\pi\rho}{\ln b - \ln a})) \right] \tag{29}$$

Let's define a quantity g_{k_1,k_2} as follows,

$$g_{k_1,k_2} \equiv \frac{1}{\ln b/a} \int_{\ln a}^{\ln b} \frac{1}{2\pi} \int_0^{2\pi} h(e^\rho,\theta) \exp(-ik_1\theta - ik_2 \frac{2\pi\rho}{\ln b/a} - \rho m) d\theta d\rho \tag{30}$$

Replace $h(e^\rho,\theta)$ in Equation (30) with the scalable Fourier-Argand representation in Equation (29), we have,

$$\begin{aligned}
g_{k_1,k_2} &= \frac{1}{\ln b/a} \int_{\ln a}^{\ln b} \frac{1}{2\pi} \int_0^{2\pi} h(e^\rho,\theta) \exp(-ik_1\theta - ik_2 \frac{2\pi\rho}{\ln b/a} - \rho m) d\theta d\rho \\
&= \frac{1}{\ln b/a} \int_{\ln a}^{\ln b} \frac{1}{2\pi} \int_0^{2\pi} \left(\sum_{t_1,t_2\in\mathbb{Z}} h_{t_1,t_2} \exp(\rho m_{t_1}) \exp(i(t_1\theta + t_2 \frac{2\pi\rho}{\ln b/a})) \right) \\
&\qquad \exp(-ik_1\theta - ik_2 \frac{2\pi\rho}{\ln b/a} - \rho m) d\theta d\rho \\
&= \frac{1}{\ln b/a} \int_{\ln a}^{\ln b} \frac{1}{2\pi} \int_0^{2\pi} \sum_{t_2\in\mathbb{Z}} \left(\sum_{t_1\in\mathbb{Z}} h_{t_1,t_2} \exp(\rho m_{t_1}) \exp(it_1\theta) \right) \exp(it_2 \frac{2\pi\rho}{\ln b/a}) \\
&\qquad \exp(-ik_1\theta) d\theta \exp(-ik_2 \frac{2\pi\rho}{\ln b/a} - \rho m) d\rho \\
&= \frac{1}{\ln b/a} \int_{\ln a}^{\ln b} \sum_{t_2\in\mathbb{Z}} \left(\sum_{t_1\in\mathbb{Z}} h_{t_1,t_2} \frac{1}{2\pi} \int_0^{2\pi} \exp(\rho m_{t_2}) \exp(i(t_1 - k_1)\theta) d\theta \right) \exp(it_2 \frac{2\pi\rho}{\ln b/a}) \\
&\qquad \exp(-ik_2 \frac{2\pi\rho}{\ln b/a} - \rho m) d\rho \\
&= \frac{1}{\ln b/a} \int_{\ln a}^{\ln b} \sum_{t_2\in\mathbb{Z}} \sum_{t_1\in\mathbb{Z}} h_{t_1,t_2} \left(\frac{1}{2\pi} \int_0^{2\pi} \exp(i(t_1 - k_1)\theta) d\theta \right) \\
&\qquad \exp(it_2 \frac{2\pi\rho}{\ln b/a}) \exp(-ik_2 \frac{2\pi\rho}{\ln b/a}) d\rho
\end{aligned} \tag{31}$$

$$
\begin{aligned}
&= \frac{1}{\ln b/a}\int_{\ln a}^{\ln b}\sum_{t_2\in\mathbb{Z}}\sum_{t_1\in\mathbb{Z}} h_{t_1,t_2}\delta[t_1-k_1]\exp(it_2\frac{2\pi\rho}{\ln b/a})\exp(-ik_2\frac{2\pi\rho}{\ln b/a})d\rho \\
&= \frac{1}{\ln b/a}\int_{\ln a}^{\ln b}\sum_{t_2\in\mathbb{Z}} h_{k_1,t_2}\exp(i(t_2-k_2)\frac{2\pi\rho}{\ln b/a})d\rho \\
&= \sum_{t_2\in\mathbb{Z}} h_{k_1,t_2}\frac{1}{\ln b/a}\int_{\ln a}^{\ln b}\exp(i(t_2-k_2)\frac{2\pi\rho}{\ln b/a})d\rho \\
&= \sum_{t_2\in\mathbb{Z}} h_{k_1,t_2}\delta[t_2-k_2] \\
&= h_{k_1,k_2}
\end{aligned}
\tag{32}
$$

The last equation holds since, when $t_2 = k_2$, then,

$$
\frac{1}{\ln b/a}\int_{\ln a}^{\ln b}\exp(i(t_2-k_2)\frac{2\pi\rho}{\ln b/a})d\rho = \frac{1}{\ln b/a}\int_{\ln a}^{\ln b}\exp(i0)dr = 1 \tag{33}
$$

And when $t_2 \neq k_2$, then

$$
\frac{1}{\ln b/a}\int_{\ln a}^{\ln b}\exp(i(t_2-k_2)\frac{2\pi\rho}{\ln b/a})d\rho = \frac{1}{\ln b/a}\int_{\ln a}^{\ln b}\exp(in\frac{2\pi\rho}{\ln b/a})d\rho = 0 \tag{34}
$$

The holds of Equation (31) proves the proposition.

B Proof of Distortion Property

Lemma B.1. *Let $T = S_s Y_t R_\alpha$, then for a transformed input $L_T[f](x)$, we can get the following property,*

$$
\begin{aligned}
\Lambda_{L_{\mathbf{T}}[f]}(\mathbf{x}) &= \Lambda_f(\mathbf{T^{-1}x})\cdot s \\
\Gamma_{L_{\mathbf{T}}[f]}(\mathbf{x}) &= \Gamma_f(\mathbf{T^{-1}x}) + \alpha
\end{aligned}
\tag{35}
$$

Proof. Let's consider $f(x)$ and $L_T[f](x)$ as the original and distroted image. Using the scalable Fourier Argand basis, we have,

$$
\begin{aligned}
\left[\Lambda_{L_T[f]}(x), \Gamma_{L_T[f]}(x)\right] &= \arg\max_{(\lambda,\gamma)} \sum_{k_1,k_2=-K}^{K} L_T[f]_{k_1,k_2}(x)\cdot c_{k_1,k_2}(\lambda,\gamma) \\
&= \arg\max_{(\lambda,\gamma)} \sum_{k_1,k_2=-K}^{K} [L_T[f] * H_{k_1,k_2}](x)\cdot\exp(-ik_1\gamma - ik_2\frac{2\pi\ln\lambda}{\ln b/a})\lambda^{-m}
\end{aligned}
\tag{36}
$$

Substitude x by $T^{-1}x$, we have,

$$
\begin{aligned}
&\left[\Lambda_{L_T[f]}(T^{-1}x), \Gamma_{L_T[f]}(T^{-1}x)\right] \\
&= \arg\max_{(\lambda,\gamma)} \sum_{k_1,k_2=-K}^{K} [f * L_{T^{-1}}[H_{k_1,k_2}]](T^{-1}x)\cdot\exp(-ik_1\gamma - ik_2\frac{2\pi\ln\lambda}{\ln b/a})\lambda^{-m}
\end{aligned}
\tag{37}
$$

Besides, from Equation (10) we have,

$$
L_T[H_{k_1,k_2}](r,\theta) = H_{k_1,k_2}(r,\theta)\cdot\left(\exp(-ik_1\alpha - ik_2\frac{2\pi\ln s}{\ln b/a})s^{-m_{k_1}}\right) \tag{38}
$$

We have,

$$
\begin{aligned}
&\left[\Lambda_{L_T[f]}(T^{-1}x), \Gamma_{L_T[f]}(T^{-1}x)\right] \\
&= \arg\max_{(\lambda,\gamma)} \sum_{k_1,k_2=-K}^{K} [f * L_{T^{-1}}[H_{k_1,k_2}]](T^{-1}x) \cdot \exp(-ik_1\gamma - ik_2 \frac{2\pi \ln \lambda}{\ln b/a})\lambda^{-m} \\
&= \arg\max_{(\lambda,\gamma)} \sum_{k_1,k_2=-K}^{K} [f * H_{k_1,k_2} \cdot \left(\exp(ik_1\alpha + ik_2 \frac{2\pi \ln s}{\ln b/a})s^{m_{k_1}}\right)](T^{-1}x) \\
&\qquad \cdot \exp(-ik_1\gamma - ik_2 \frac{2\pi \ln \lambda}{\ln b/a})\lambda^{-m} \\
&= \arg\max_{(\lambda,\gamma)} \sum_{k_1,k_2=-K}^{K} [f * H_{k_1,k_2}](T^{-1}x) \cdot \exp(-ik_1(\gamma-\alpha) - ik_2 \frac{2\pi \ln(\lambda/s)}{\ln b/a})(\lambda/s)^{-m} \\
&= \arg\max_{(\lambda s,\gamma+\alpha)} \sum_{k_1,k_2=-K}^{K} [f * H_{k_1,k_2}](T^{-1}x) \cdot \exp(-ik_1(\gamma) - ik_2 \frac{2\pi \ln(\lambda)}{\ln b/a})(\lambda)^{-m}
\end{aligned}
\tag{39}
$$

Finally, we can conclude that,

$$
\begin{aligned}
\Lambda_{L_{\mathbf{T}}[f]}(\mathbf{x}) &= \Lambda_f(\mathbf{T^{-1}x}) \cdot s \\
\Gamma_{L_{\mathbf{T}}[f]}(\mathbf{x}) &= \Gamma_f(\mathbf{T^{-1}x}) + \alpha
\end{aligned}
\tag{40}
$$

□

C Details, Analysis, and Visualization

C.1 Additional Studies

Stability We first quantify the deformation stability, and seek to answer in what degree of distortion can the method persist the equivariant (or the generalization ability) compared with original convolution. To achieve this, we evalue our method on the test dataset that rotated and scaled entirely by a certain amount. We do this experiment with the MNIST dataset. Figure 4 displays the accuracy decay with different settings. It can be observed that when rotate the image that largely differ from the training dataset, the CNNs will drop accuracy largely, while our method maintains a consistently excellent performance. For scale changing test, our method maintain a relatively good capability on a wide range of scale change. For extremely large scale change, our method have performance drop due to limited filter spot and sampling approximation.

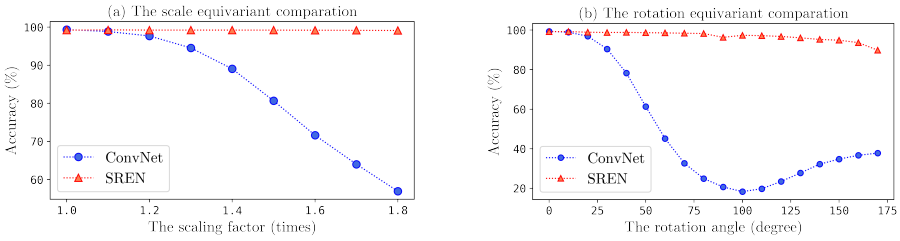

Figure 4: Equivariance Stability: When the entire test set is translated and rotated by a certain scale or angle, our network can still get good performance, while the ConvNet's performance will decrease largely with the change.

C.2 Feature visualization

Besides numerical experiments, we also visualize the features of the network to verify the equivariance achieved by our network more intuitively. The visualization result is shown inFigure 5. This visualization answer the question whether the hidden feature visually looks equivariant with respect

to rotation and scaling. The parameters are randomly initialized for both CNN and SREN model. With the compensation view, we can clearly see that our network indeed achieve the equivariant in rotation and scaling.

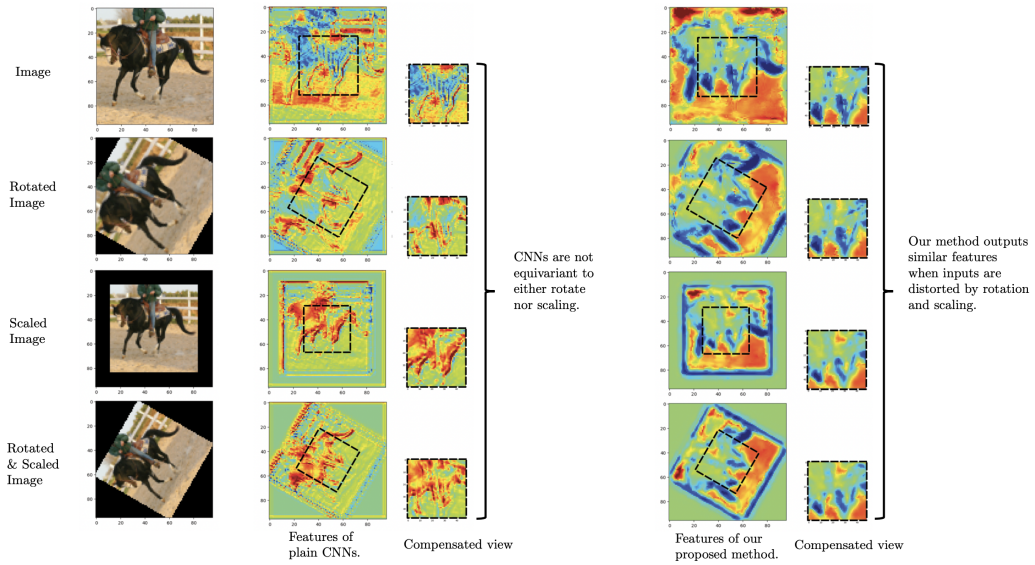

Figure 5: Feature Visualization: We visualize the output feature of the CNNs and our SREN in compensating view. It can be seen that the features of our method do not vary with the transformation of the object, compared with CNNs.

C.3 ARCHITECTURE AND PIPELINE

The main flow of our algorithm is shown in Figure 2. This method can be applied to a multi-layer network structure. Assuming that the network contains K blocks (which we named as SimBlock), where each block contains multiple convolutional layers. And the input signal of the i-th block is denoted as f_i. We first calculate the scalable Fourier Argand feature proposed in Section 4.1. This allows us to get a spatial-wised matrix $\mathbf{M}_{f_i}(\mathbf{x})$. Then all the SimConv layer (proposed in Section 4.2) within the SimBlock share this same scalable Fourier Argand feature. Let's assume there are N SimConv layers in each SimBlock. Based on the property of Equation (21), the output feature of this block, denoted as f_{i+1}, is guaranteed to be an equivariant feature corresponding to the input image. For the classification task, the invariance property is desired. We pooled the last layer for each channel, followed by the MLP layer. The final output is then obtained and guaranteed invariant for any similar transform. The whole process is shown in Algorithm 1.

C.4 STL-10 DATASET COMPARATION

This section compare our method with other related methods with the same backbone. All method use WideResNet as the backbone with 16 layers and a widen factor as 8. We trained our network for 1000 epochs with a batch size of 128, we use SGD as the optimizer. The initial learning rate is set to 0.1 and decreased by a factor of 0.2 after 300 epoch, and the drop rate probability as 0.3. We ues the data augmentation technique following Sosnovik et al. (2021a). Table 3 demostrates our main results compared to recent baselines, such as SiCNN (Kanazawa et al., 2014), DSS (Worrall & Welling, 2019), SESN (Sosnovik et al., 2020), DISCO (Sosnovik et al., 2021a). Our method obtained a competitive result that is close to the state-of-the-art paper. Besides, we want to highlight that our method's computation cost and parameters are similar to the classic convolutional neural network with the same backbone. This is because the two quantities Λ and Γ can be shared within different filters. Thus the cost is small compared with the abundance of the convolutional (or our SimConv) layer. And the complexity of our SimConv operator is similar to the convolutional layer.

Algorithm 1: Pipeline of our SREN Algorithm

Input: Batch of data $\{f_i\}_{i=0}^{T}$
Output: predicted output value v
for $i = 1 : K$ *levels* **do**
  Initialize filter h, calculate its basis H_{k_1,k_2};
  Take f_i as input;
  Get $f_{k_1,k_2} \leftarrow f \star H_{k_1,k_2}$;
  Get $\Gamma_{f_i}, \lambda_{f_i} \leftarrow \arg\max_{\gamma,\lambda} \mathbf{c}_{\gamma,\lambda}\mathbf{F}$;
  Calculate $M_f \leftarrow \Gamma_{f_i}, \lambda_{f_i}$;
  for $m = 1 : M$ *layers in specific level n* **do**
    Apply M_f to SimConv;
    $f_{i,m+1} \leftarrow f_i \circledast \varphi_{i,m}$;
  $f_{i+1} \leftarrow f_{i,M}$
Get the output with a head layer: $v \leftarrow \text{MLP}(f)$

Table 3: Methods comparation on STL-10 dataset, all methods use the WideResNet as the backbone.

Method	WRN	SiCNN	SI-ConvNet	DSS	SS-CNN	SESN	DISCO	SREN
Error	11.48	11.62	12.48	11.28	25.47	8.51	8.07	8.23

For comparison, the computational cost of DISCO method takes more than 5 times longer and SESN takes more than 16.5 times longer than classic CNNs, reported in Sosnovik et al. (2021a).