Journal of Machine Learning Research 23 (2022) 1-45 Submitted 2/20; Published 2/22

Scaling-Translation-Equivariant Networks with Decomposed Convolutional Filters

Wei Zhu ZHU@MATH.UMASS.EDU
Department of Mathematics and Statistics
University of Massachusetts Amherst
Amherst, MA 01003, USA

Qiang Qiu QIANG.QIU@DUKE.EDU
Department of Electrical and Computer Engineering
Purdue University
West Lafayette, Indiana 47907, USA

Robert Calderbank ROBERT.CALDERBANK@DUKE.EDU
Guillermo Sapiro GUILLERMO.SAPIRO@DUKE.EDU
Department of Electrical and Computer Engineering
Duke University
Durham, NC 27708, USA

Xiuyuan Cheng XIUYUAN.CHENG@DUKE.EDU
Department of Mathematics
Duke University
Durham, NC 27708, USA

Editor: Honglak Lee

Abstract

Encoding the scale information explicitly into the representation learned by a convolutional neural network (CNN) is beneficial for many computer vision tasks especially when dealing with multiscale inputs. We study, in this paper, a scaling-translation-equivariant ($\mathcal{ST}$-equivariant) CNN with joint convolutions across the space and the scaling group, which is shown to be both sufficient and necessary to achieve equivariance for the regular representation of the scaling-translation group $\mathcal{ST}$. To reduce the model complexity and computational burden, we decompose the convolutional filters under two pre-fixed separable bases and truncate the expansion to low-frequency components. A further benefit of the truncated filter expansion is the improved deformation robustness of the equivariant representation, a property which is theoretically analyzed and empirically verified. Numerical experiments demonstrate that the proposed scaling-translation-equivariant network with decomposed convolutional filters (ScDCFNet) achieves significantly improved performance in multiscale image classification and better interpretability than regular CNNs at a reduced model size.

Keywords: convolutional neural network (CNN), computer vision, scaling-translation-equivariant, decomposed convolutional filters, deformation robust equivariant representation

1. Introduction

Convolutional neural networks (CNNs) have achieved great success in machine learning problems such as image classification (Krizhevsky et al., 2012), object detection (Ren et al., 2015), and semantic segmentation (Long et al., 2015; Ronneberger et al., 2015). Compared to fully-connected networks, CNNs through spatial weight sharing have the benefit of being translation-equivariant, i.e., translating the input leads to a translated version of the output. This property is crucial for many vision tasks such as image recognition and segmentation. However, regular CNNs are not equivariant to other important group transformations such as rescaling and rotation, and it is beneficial in some applications to also encode such group information explicitly into the network representation.

Several network architectures have been designed to achieve (2D) roto-translation-equivariance ($SE(2)$-equivariance) (Weiler and Cesa, 2019; Cheng et al., 2019; Hoogeboom et al., 2018; Worrall et al., 2017; Bekkers et al., 2017; Zhou et al., 2017; Marcos et al., 2017; Weiler et al., 2018b), i.e., roughly speaking, if the input is spatially rotated and translated, the output is transformed accordingly. The feature maps of such networks typically include an extra index for the rotation group $SO(2)$. Building on the idea of group convolutions proposed by Cohen and Welling (2016) for discrete symmetry groups, Cheng et al. (2019), Worrall et al. (2017), and Weiler et al. (2018b) constructed $SE(2)$-equivariant CNNs by conducting group convolutions jointly across the space and $SO(2)$ using steerable filters (Freeman and Adelson, 1991).

Scaling-translation-equivariant ($\mathcal{ST}$-equivariant) CNNs, on the other hand, are usually studied in a less general setting in the existing literature. In particular, joint convolutions across the space and the scaling group $\mathcal{S}$ are typically not proposed to achieve equivariance in the general form (Kanazawa et al., 2014; Marcos et al., 2018; Xu et al., 2014; Ghosh and Gupta, 2019). This is possibly because of two difficulties one encounters when dealing with the scaling group: First, unlike $SO(2)$, it is an acyclic and unbounded group; second, an extra index in $\mathcal{S}$ incurs a significant increase in model parameters and computational burden. Moreover, due to changing view angle or numerical discretization, the scaling transformation is rarely perfect in practice. One thus needs to quantify and promote the deformation robustness of the equivariant representation (i.e., is the model still "approximately" equivariant if the scaling transformation is "contaminated" by a nuisance input deformation), which, to the best of our knowledge, has not been studied in prior works.

The purpose of this paper is to address the aforementioned theoretical and practical issues in the construction of $\mathcal{ST}$-equivariant CNN models. Specifically, our contribution is three-fold:

1. We propose a general $\mathcal{ST}$-equivariant CNN architecture with a joint convolution over $\mathbb{R}^2$ and $\mathcal{S}$, which is proved in Section 4 to be both sufficient and necessary to achieve equivariance for the regular representation of the group $\mathcal{ST}$.

2. A truncated decomposition of the convolutional filters under a pre-fixed separable basis on the two geometric domains ($\mathbb{R}^2$ and $\mathcal{S}$) is used to reduce the model size and computational cost.

3. We prove the representation stability of the proposed architecture up to equivariant scaling action of the input signal, guaranteeing equivariance is "approximately" achieved

even if the scaling effect is non-perfect. This theoretical result is crucial for the practical implementation of $\mathcal{ST}$-equivariant CNNs when signals are discrete and confined in finite domains.

Our contribution to the family of group-equivariant CNNs is non-trivial; in particular, the scaling group unlike $SO(2)$ is acyclic and non-compact. This poses challenges both in theory and in practice, so that many previous works on group-equivariant CNNs cannot be directly extended. The concurrent independent research by Worrall and Welling (2019), Sosnovik et al. (2020), and Bekkers (2020) also discovered and implemented joint convolutions to achieve scaling-translation-equivariance in the general form. However, the approach by Worrall and Welling (2019) is limited to scaling factors at only integer powers of 2, and none of the three analyzes and promotes the deformation robustness of the equivariant representation, especially in the practical setting where signals are discretized and computed only on compact domains. We introduce new algorithm design and mathematical techniques to obtain general $\mathcal{ST}$-equivariant CNNs with both computational efficiency and proved representation stability.

2. Related Work

Mixed-scale CNNs. Incorporating multiscale information into a CNN representation has been studied in many existing works. The Inception net (Szegedy et al., 2015) and its generalizations (Szegedy et al., 2017, 2016; Li et al., 2019) stack filters of different sizes in a single layer to address the multiscale salient features. Dilated convolutions (Pelt and Sethian, 2018; Wang et al., 2018; Yu and Koltun, 2016; Yu et al., 2017), pyramid architectures (Ke et al., 2017; Lin et al., 2017), and multiscale dense networks (Huang et al., 2018) have also been proposed to take into account the multiscale feature information. Although the effectiveness of such models have been empirically demonstrated in various vision tasks, there is still a lack of interpretability of their ability to encode the input scale information.

Group-equivariant CNNs. Group-equivariant CNNs (G-CNNs) have consistently demonstrated their superior performance over classical CNNs by explicitly encoding group information into the network representation. G-CNN was initially proposed by Cohen and Welling (2016) to achieve equivariance over finite discrete symmetry groups. The idea is later generalized in (Cohen et al., 2019; Kondor and Trivedi, 2018; Cohen and Welling, 2017), and has been applied mainly to discrete or compact continuous groups such as 2D rotation $SO(2)$ (Weiler and Cesa, 2019; Cheng et al., 2019; Hoogeboom et al., 2018; Worrall et al., 2017; Bekkers et al., 2017; Zhou et al., 2017; Marcos et al., 2017; Weiler et al., 2018b) and 3D rotation $SO(3)$ (Weiler et al., 2018a; Worrall and Brostow, 2018; Thomas et al., 2018; Cohen et al., 2018; Esteves et al., 2018; Winkels and Cohen, 2018; Andrearczyk et al., 2019).

Although $\mathcal{ST}$-equivariant (or invariant) CNNs have also been proposed in the literature (Kanazawa et al., 2014; Marcos et al., 2018; Xu et al., 2014; Ghosh and Gupta, 2019), they are typically studied in a less general setting. In particular, none of these works proposed to conduct joint convolutions over $\mathbb{R}^2 \times \mathcal{S}$ as a necessary and sufficient condition to achieve equivariance for the regular representation of the group $\mathcal{ST}$, for which reason they are thus variants of a special case of our proposed architecture where the convolutional filters in $\mathcal{S}$ are Dirac delta functions (c.f. Remark 1.) The interscale convolution proposed in the independent concurrent works by Worrall and Welling (2019), Sosnovik et al. (2020), and Bekkers (2020)

bear the most resemblance to our proposed model. In particular, the filter expansion under B-splines for arbitrary Lie groups proposed by Bekkers (2020) is also akin to our truncated filter decomposition under compactly supported separable function bases. However, the approach by Worrall and Welling (2019) is limited to scaling factors at only integer powers of 2, and none of the three concurrent works analyzes and promotes deformation robustness of the equivariant representation, which is important both in theory and in practice because scaling effect is rarely perfect due to signal distortion, discretization, and truncation.

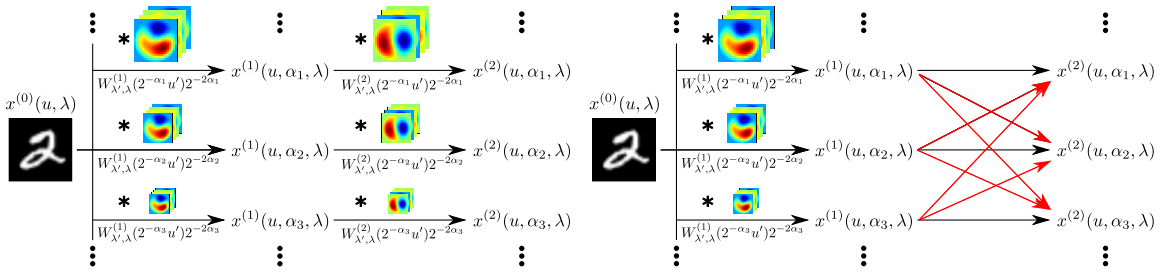
Representation stability to input deformations. Input deformations typically induce noticeable variabilities within object classes, some of which are uninformative for the vision tasks. Models that are stable to input deformations are thus favorable in many applications. The scattering transform (Bruna and Mallat, 2013; Mallat, 2010, 2012) computes translation-invariant representations that are Lipschitz continuous to deformations by cascading predefined wavelet transforms and modulus poolings. A joint convolution over $\mathbb{R}^2 \times SO(2)$ is later adopted by Sifre and Mallat (2013) to build roto-translation scattering with stable rotation/translation-invariant representations. These models, however, use pre-fixed wavelet transforms in the networks, and are thus nonadaptive to the data. Invariance and stability of deep convolutional representations have also be studied by Bietti and Mairal (2017, 2019) in the context of convolutional kernel networks (Mairal, 2016; Mairal et al., 2014). DCFNet (Qiu et al., 2018) combines a pre-fixed filter basis and learnable expansion coefficients in a CNN architecture, achieving both data adaptivity and representation stability inherited from the filter regularity. This idea is later extended by Cheng et al. (2019) to produce $SE(2)$-equivariant representations that are Lipschitz continuous in L^2 norm to input deformations modulo a global rotation, i.e., the model stays approximately equivariant even if the input rotation is imperfect. To the best of our knowledge, a theoretical analysis of the deformation robustness of a $\mathcal{ST}$-equivariant CNN has yet been studied, and a direct generalization of the result by Cheng et al. (2019) is futile because the feature maps of a $\mathcal{ST}$-equivariant CNN is typically not in L^2 (c.f. Remark 4.)

3. $\mathcal{ST}$-Equivariant CNN And Filter Decomposition

Group-equivariance is the property of a mapping $f : X \to Y$ to commute with the group actions on the domain X and codomain Y. More specifically, let G be a group, and D_g, T_g, respectively, be group actions on X and Y. A function $f : X \to Y$ is said to be G-equivariant if

$$f(D_g x) = T_g(f(x)), \quad \forall\ g \in G,\ x \in X. \tag{1}$$

G-invariance is thus a special case of G-equivariance where $T_g = \mathrm{Id}_Y$. For learning tasks where the feature $y \in Y$ is known a priori to change equivariantly to a group action $g \in G$ on the input $x \in X$, e.g. image segmentation should be equivariant to translation, it would be beneficial to reduce the hypothesis space to include only G-equivariant models. In this paper, we consider mainly the scaling-translation group $\mathcal{ST} = \mathcal{S} \ltimes \mathbb{R}^2 \cong \mathbb{R} \times \mathbb{R}^2$, which is the semi-direct product between the the scaling group $\mathcal{S}$ and the translation group $\mathbb{R}^2$. Given $g = (\beta, v) \in \mathcal{ST}$ and an input image $x^{(0)}(u, \lambda)$ ($u \in \mathbb{R}^2$ is the spatial position, and λ is the unstructured channel index, e.g. RGB channels of a color image), the scaling-translation



(a) A special case of $\mathcal{ST}$-equivariant CNN with (multiscale) spatial convolutions.

(b) The general case of $\mathcal{ST}$-equivariant CNN with joint convolutions (Theorem 1).

Figure 1: (a) A special case of $\mathcal{ST}$-equivariant CNN with only (multiscale) spatial convolutions. The previous works on $\mathcal{ST}$-equivariant CNNs (Kanazawa et al., 2014; Marcos et al., 2018; Xu et al., 2014; Ghosh and Gupta, 2019) are all variants of this architecture. (b) The general case of $\mathcal{ST}$-equivariant CNN with joint convolutions (Theorem 1) where information transfers among different scales. See Remark 1 for more explanation.

group action $D_g = D_{\beta,v}$ on $x^{(0)}$ is defined as

$$D_{\beta,v}x^{(0)}(u,\lambda) \coloneqq x^{(0)}\left(2^{-\beta}(u-v),\lambda\right). \tag{2}$$

Constructing $\mathcal{ST}$-equivariant CNNs thus amounts to finding an architecture $\mathcal{A}$ such that each trained network $f \in \mathcal{A}$ commutes with the group action $D_{\beta,v}$ on the input and a similarly defined group action $T_{\beta,v}$ (to be explained in Section 3.1) on the output.

3.1 $\mathcal{ST}$-Equivariant CNNs

Inspired by Cheng et al. (2019), Weiler et al. (2018b), and Cohen et al. (2019), we consider $\mathcal{ST}$-equivariant CNNs with an extra index $\alpha \in \mathcal{S}$ for the the scaling group $\mathcal{S} \cong \mathbb{R}$: for each $l \geq 1$, the l-th layer output is denoted as $x^{(l)}(u,\alpha,\lambda)$, where $u \in \mathbb{R}^2$ is the spatial position, $\alpha \in \mathcal{S}$ is the scale index, and $\lambda \in [M_l] \coloneqq \{1,\dots,M_l\}$ corresponds to the unstructured channels. We use the continuous model for formal derivation, i.e., the images and feature maps have continuous spatial and scale indices. In practice, the images are discretized on a Cartesian grid, and the scales are computed only on a discretized finite interval (c.f. Section 5.) We define the group action $T_{\beta,v}$ on the l-th layer output as a scaling-translation in space as well as a shift in the scale channel, which corresponds to the regular representation of the group $\mathcal{ST}$ on the space of l-th layer feature maps (Cohen and Welling, 2016; Cohen et al., 2019):

$$T_{\beta,v}x^{(l)}(u,\alpha,\lambda) \coloneqq x^{(l)}\left(2^{-\beta}(u-v),\alpha-\beta,\lambda\right), \quad \forall\, l \geq 1. \tag{3}$$

A feedforward neural network is said to be $\mathcal{ST}$-equivariant (under the group actions $D_{\beta,v}$ and $T_{\beta,v}$) if

$$x^{(l)}[D_{\beta,v}x^{(0)}] = T_{\beta,v}x^{(l)}[x^{(0)}], \quad \forall\, l \geq 1, \tag{4}$$

where we slightly abuse the notation $x^{(l)}[x^{(0)}]$ to denote the l-th layer output given the input $x^{(0)}$. The following Theorem shows that $\mathcal{ST}$-equivariance (4) is achieved if and only if joint convolutions are conducted over $\mathcal{S}\times\mathbb{R}^2$ as in (5) and (6).

Theorem 1. *A feedforward neural network with an extra index $\alpha \in \mathcal{S}$ for layerwise output is $\mathcal{ST}$-equivariant (under the group actions $D_{\beta,v}$ and $T_{\beta,v}$) if and only if the layerwise operations are defined as* (5) *and* (6)*:*

$$x^{(1)}[x^{(0)}](u,\alpha,\lambda) = \sigma\left(\sum_{\lambda'}\int_{\mathbb{R}^2} 2^{-2\alpha}x^{(0)}(u+u',\lambda')W^{(1)}_{\lambda',\lambda}\left(2^{-\alpha}u'\right)du' + b^{(1)}(\lambda)\right), \tag{5}$$

$$x^{(l)}[x^{(l-1)}](u,\alpha,\lambda) = \sigma\left(\sum_{\lambda'}\int_{\mathbb{R}^2}\int_{\mathbb{R}} 2^{-2\alpha}x^{(l-1)}(u+u',\alpha+\alpha',\lambda')\cdot W^{(l)}_{\lambda',\lambda}\left(2^{-\alpha}u',\alpha'\right)d\alpha' du' + b^{(l)}(\lambda)\right), \quad \forall l>1, \tag{6}$$

where $\sigma:\mathbb{R}\to\mathbb{R}$ is a pointwise nonlinear function, $W^{(1)}_{\lambda',\lambda}(u)$ is the spatial convolutional filter in the first layer with output channel λ and input channel λ', and $W^{(l)}_{\lambda',\lambda}(u,\alpha)$ is the space-scale joint convolutional filter for layer $l>1$.

We defer the proof of Theorem 1, as well as those of other theorems, to the appendix. We note that the joint-convolution in Theorem 1 is a generalization of the group convolution proposed by Cohen and Welling (2016) to a non-compact group $\mathcal{ST}$ in the continuous setting, which is known to be necessary and sufficient to achieve equivariance under regular representations of the group (Cohen et al., 2019).

Remark 1. *When the convolutional filter $W^{(l)}_{\lambda',\lambda}(u,\alpha)$ takes the special form $W^{(l)}_{\lambda',\lambda}(u,\alpha) = V^{(l)}_{\lambda',\lambda}(u)\delta(\alpha)$, where δ is the Dirac delta function, the joint convolution* (6) *over $\mathbb{R}^2\times\mathcal{S}$ reduces to only a (multiscale) spatial convolution*

$$x^{(l)}[x^{(l-1)}](u,\alpha,\lambda) = \sigma\left(\sum_{\lambda'}\int_{\mathbb{R}^2} x^{(l-1)}(u+u',\alpha,\lambda')V^{(l)}_{\lambda',\lambda}\left(2^{-\alpha}u'\right)2^{-2\alpha}du' + b^{(l)}(\lambda)\right), \tag{7}$$

i.e., the feature maps at different scales do not transfer information among each other (see Figure 1a). The previous works (Kanazawa et al., 2014; Marcos et al., 2018; Xu et al., 2014; Ghosh and Gupta, 2019) on $\mathcal{ST}$-equivariant CNNs are all based on this special case of Theorem 1.

Although the joint convolutions (6) on $\mathbb{R}^2\times\mathcal{S}$ provide the most general way of imposing $\mathcal{ST}$-equivariance under $D_{\beta,v}$ and $T_{\beta,v}$, they unfortunately also incur a significant increase in the model size and computational burden. Following the idea of Cheng et al. (2019) and Qiu et al. (2018), we address this issue by taking a truncated decomposition of the convolutional filters under a pre-fixed separable basis, which will be discussed in detail in the next section.

3.2 Separable Basis Decomposition

We consider decomposing the convolutional filters $W^{(l)}_{\lambda',\lambda}(u,\alpha)$ under the product of two orthogonal function bases, $\{\psi_k(u)\}_k$ and $\{\varphi_m(\alpha)\}_m$, which are the eigenfunctions of the Dirichlet Laplacian on, respectively, the rectangle $D=[-1,1]^2\subset\mathbb{R}^2$ and $I_\alpha=[-1,1]$, i.e.,

$$\begin{cases}\Delta\psi_k = -\mu_k\psi_k & \text{in } D,\\ \psi_k = 0 & \text{on } \partial D,\end{cases} \quad \text{and} \quad \begin{cases}\varphi_m'' = -\nu_m\varphi_m & \text{in } I_\alpha=[-1,1]\\ \varphi_m(-1)=\varphi_m(1)=0.\end{cases} \tag{8}$$

Remark 2. *The choice of the spatial function basis $\{\psi_k(u)\}_k$ is not unique. For example, one can also choose $\psi_k(u)$ to be the eigenfunctions of the Dirichlet Laplacian on the unit disk, i.e., the Fourier-Bessel basis (Abramowitz and Stegun, 1965) considered by Cheng et al. (2019). The only requirement for $\{\psi_k(u)\}_k$ is that it is an orthogonal basis vanishing on the boundary of the domain—a property we will use later in the proof of deformation stability of the equivariant representation (Theorem 2) in Section 4. Empirically, we found $\psi_k(u) = \psi_k(u_1, u_2)$ defined in (8), which is separable in the two spatial coordinates, to consistently achieve better results.*

Compared to $\psi_k(u)$, the function basis in scale $\{\varphi_m(\alpha)\}_m$ is less restrictive, as the choice of which does not contribute to the stability analysis for spatial deformation in Theorem 2. However, as the joint convolutional filters $W^{(l)}_{\lambda',\lambda}(u,\alpha)$ are assumed to be compactly supported in both space and scale (otherwise it is hard to implement in practice), it is natural to choose a compactly supported orthogonal function basis $\{\varphi_m(\alpha)\}_m$ for scale.

In the continuous formulation, the spatial "pooling" operation is equivalent to rescaling the convolutional filters in space. We thus assume, without loss of generality, that the convolutional filters are compactly supported on a rescaled domain as follows

$$W^{(1)}_{\lambda',\lambda} \in C_c(2^{j_1}D), \quad \text{and} \quad W^{(l)}_{\lambda',\lambda} \in C_c(2^{j_l}D \times I_\alpha), \quad \forall\, l > 1.$$

Let $\psi_{j,k}(u) \coloneqq 2^{-2j}\psi_k(2^{-j}u)$, then $W^{(l)}_{\lambda',\lambda}$ can be decomposed under $\{\psi_{j_l,k}\}_k$ and $\{\varphi_m\}_m$ as

$$W^{(1)}_{\lambda',\lambda}(u) = \sum_k a^{(1)}_{\lambda',\lambda}(k)\psi_{j_1,k}(u),\ W^{(l)}_{\lambda',\lambda}(u,\alpha) = \sum_m \sum_k a^{(l)}_{\lambda',\lambda}(k,m)\psi_{j_l,k}(u)\varphi_m(\alpha),\ l > 1 \quad (9)$$

where $a^{(1)}_{\lambda',\lambda}(k)$ and $a^{(l)}_{\lambda',\lambda}(k,m)$ are the expansion coefficients of the filters. During training, the basis functions are fixed, and only the expansion coefficients are updated. In practice, we truncate the expansion to only low-frequency components (i.e., $a^{(l)}_{\lambda',\lambda}(k,m)$ are non-zero only for $k \in [K]$, $m \in [K_\alpha]$), which are kept as the trainable parameters. Similar idea has also been considered in the prior works (Qiu et al., 2018; Cheng et al., 2019; Jacobsen et al., 2016). Since ψ_k and φ_m are the separable eigenfunctions of the Dirichlet Laplacian, standard convergence results for generalized Fourier series apply, e.g., if $W^{(l)}_{\lambda',\lambda} \in C^p(D \times I_\alpha)$, then the truncated expansion converges uniformly to $W^{(l)}_{\lambda',\lambda}$ at rate $O(\log(KK_\alpha)/(K^pK^p_\alpha))$ (Jackson, 1930). We call the resulting model Scaling-translation-equivariant Network with Decomposed Convolutional Filters (ScDCFNet).

Truncating the filter expansion leads directly to a reduction of network parameters and computational burden. More specifically, let us compare the l-th convolutional layer (6) of a $\mathcal{ST}$-equivariant CNN with and without truncated basis decomposition:

Number of trainable parameters: Suppose the filters $W^{(l)}_{\lambda',\lambda}(u,\alpha)$ are discretized on a Cartesian grid of size $L \times L \times L_\alpha$. The number of trainable parameters at the l-th layer of an $\mathcal{ST}$-equivariant CNN without basis decomposition is $L^2L_\alpha M_{l-1}M_l$. On the other hand, in an ScDCFNet with truncated basis expansion up to K leading coefficients for u and K_α coefficients for α, the number of parameters is instead $KK_\alpha M_{l-1}M_l$. Hence a reduction to a factor of KK_α/L^2L_α in trainable parameters is achieved for ScDCFNet via truncated basis decomposition. In particular, if $L = 5, L_\alpha = 5, K = 8$, and $K_\alpha = 3$, then the number of parameters is reduced to $(8 \times 3)/(5^2 \times 5) = 19.2\%$.

Remark 3. *We want to point out that even though the number of trainable parameters has been reduced after truncated expansion, the memory usage of the entire network remains the same. The reason is that the bottleneck of memory consumption in a deep network is the storage of the feature maps instead of the trainable weights.*

Computational cost: Suppose the size of the input $x^{(l-1)}(u,\alpha,\lambda)$ and output $x^{(l)}(u,\alpha,\lambda)$ at the l-th layer are, respectively, $H\times W\times N_s\times M_{l-1}$ and $H\times W\times N_s\times M_l$, where $H\times W$ is the spatial dimension, N_s is the number of scale channels, and $M_{l-1}(M_l)$ is the number of the unstructured input (output) channels. Let the filters $W^{(l)}_{\lambda',\lambda}(u,\alpha)$ be discretized on a Cartesian grid of size $L\times L\times L_\alpha$. The following proposition shows that, compared to a regular $\mathcal{ST}$-equivariant CNN, the computational cost in a forward pass of ScDCFNet is reduced again to a factor of KK_α/L^2L_α.

Proposition 1. *Assume $M_l\gg L^2, L_\alpha$, i.e., the number of the output channels is much larger than the size of the convolutional filters in u and α, then the computational cost of an ScDCFNet is reduced to a factor of KK_α/L^2L_α when compared to a $\mathcal{ST}$-equivariant CNN without basis decomposition.*

4. Representation Stability of ScDCFNet to Input Deformation

Apart from reducing the model size and computational burden, we demonstrate in this section that truncating the filter decomposition has the further benefit of improving deformation robustness of the equivariant representation, i.e., the equivariance relation (4) still approximately holds true even if the spatial scaling of the input $D_{\beta,v}x^{(0)}$ is contaminated by a local deformation. The analysis is motivated by the fact that scaling transformations are rarely perfect in practice—they are typically subject to local distortions such as changing view angle or numerical discretization. To quantify the distance between different feature maps at each layer, we define the norm of $x^{(l)}$ as

$$\|x^{(0)}\|^2=\frac{1}{M_0}\sum_{\lambda=1}^{M_0}\int\left|x^{(0)}(u,\lambda)\right|^2du,\quad \|x^{(l)}\|^2=\sup_{\alpha\in\mathbb{R}}\frac{1}{M_l}\sum_{\lambda=1}^{M_l}\int\left|x^{(l)}(u,\alpha,\lambda)\right|^2du,\ l\geq 1. \tag{10}$$

Remark 4. *The definition of $\|x^{(l)}\|$ is different from that of RotDCFNet (Cheng et al., 2019), where an L^2 norm is taken for the α index as well. The reason why we adopt the L^∞ norm for α in* (10) *is that $x^{(l)}$ is typically not L^2 in α, since the scaling group $\mathcal{S}$, unlike $SO(2)$, has infinite Haar measure.*

We next quantify the representation stability of ScDCFNet under three mild assumptions on the convolutional layers and input deformations. First,

(A1) The pointwise nonlinear activation $\sigma:\mathbb{R}\to\mathbb{R}$ is non-expansive, i.e., $|\sigma(x)-\sigma(y)|\leq|x-y|$. For example, the rectified linear unit (ReLU) satisfies this property.

Next, we need a bound on the convolutional filters under certain norms. For each $l \geq 1$, define A_l as

$$\begin{cases} A_1 \coloneqq \pi \max\left\{ \sup_{\lambda} \sum_{\lambda'=1}^{M_0} \|a^{(1)}_{\lambda',\lambda}\|_\mu,\ \frac{M_0}{M_1} \sup_{\lambda'} \sum_{\lambda=1}^{M_1} \|a^{(1)}_{\lambda',\lambda}\|_\mu \right\}, \\ A_l \coloneqq \pi \max\left\{ \sup_{\lambda} \sum_{\lambda'=1}^{M_{l-1}} \sum_{m} \|a^{(l)}_{\lambda',\lambda}(\cdot,m)\|_\mu,\ \frac{2M_{l-1}}{M_l} \sum_{m} \sup_{\lambda'} \sum_{\lambda=1}^{M_l} \|a^{(l)}_{\lambda',\lambda}(\cdot,m)\|_\mu \right\}, \end{cases} \tag{11}$$

where the weighted l^2-norm $\|a\|_\mu$ of a sequence $\{a(k)\}_{k\geq 0}$ is defined as $\|a\|_\mu^2 \coloneqq \sum_k \mu_k a(k)^2$, where μ_k is the k-th eigenvalue of the Dirichlet Laplacian on $[-1,1]^2$ defined in (8). We next assume that each A_l is bounded:

(A2) For all $l \geq 1$, $A_l \leq 1$.

The boundedness of A_l is facilitated by truncating the basis decomposition to only low-frequency components (small μ_k), which is one of the key idea of ScDCFNet explained in Section 3.2. After a proper initialization of the trainable coefficients, (A2) can generally be satisfied. The assumption (A2) implies several bounds on the convolutional filters at each layer (c.f. Lemma 2 in the appendix), which, combined with (A1), guarantees that an ScDCFNet is layerwise non-expansive:

Proposition 2. *Under the assumption (A1) and (A2), an ScDCFNet satisfies the following.*

(a) For any $l \geq 1$, the mapping of the l-th layer, $x^{(l)}[\cdot]$ defined in (5) and (6), is non-expansive, i.e.,

$$\|x^{(l)}[x_1] - x^{(l)}[x_2]\| \leq \|x_1 - x_2\|, \quad \forall x_1, x_2.$$

(b) Let $x_0^{(l)}$ be the l-th layer output given a zero bottom-layer input, then $x_0^{(l)}(\lambda)$ depends only on λ.

(c) Let $x_c^{(l)}$ be the centered version of $x^{(l)}$ after removing $x_0^{(l)}$, i.e., $x_c^{(0)}(u,\lambda) \coloneqq x^{(0)}(u,\lambda) - x_0^{(0)}(\lambda) = x^{(0)}(u,\lambda)$, and $x_c^{(l)}(u,\alpha,\lambda) \coloneqq x^{(l)}(u,\alpha,\lambda) - x_0^{(l)}(\lambda)$, $\forall l \geq 1$, then $\|x_c^{(l)}\| \leq \|x_c^{(l-1)}\|$, $\forall l \geq 1$. As a result, $\|x_c^{(l)}\| \leq \|x_c^{(0)}\| = \|x^{(0)}\|$.

Finally, we make an assumption on the input deformation modulo a global scale change. Given a C^2 function $\tau : \mathbb{R}^2 \to \mathbb{R}^2$, the spatial deformation D_τ on the feature maps $x^{(l)}$ is defined as

$$D_\tau x^{(0)}(u,\lambda) = x^{(0)}(\rho(u),\lambda), \quad \text{and} \quad D_\tau x^{(l)}(u,\alpha,\lambda) = x^{(l)}(\rho(u),\alpha,\lambda), \quad l \geq 1, \tag{12}$$

where $\rho(u) = u - \tau(u)$. We assume a small local deformation on the input:

(A3) $|\nabla\tau|_\infty \coloneqq \sup_u \|\nabla\tau(u)\| < 1/5$, where $\|\cdot\|$ is the operator norm.

The following theorem demonstrates the representation stability of an ScDCFNet to input deformation modulo a global scale change.

Theorem 2. *Let D_τ be a small spatial deformation defined in* (12)*, and let $D_{\beta,v}, T_{\beta,v}$ be the group actions corresponding to an arbitrary scaling $2^{-\beta} \in \mathbb{R}_+$ centered at $v \in \mathbb{R}^2$ defined in* (2) *and* (3)*. In an ScDCFNet satisfying (A1), (A2), and (A3), we have, for any L,*

$$\left\| x^{(L)}[D_{\beta,v} \circ D_\tau x^{(0)}] - T_{\beta,v} x^{(L)}[x^{(0)}] \right\| \leq 2^{\beta+1} \left(4L|\nabla \tau|_\infty + 2^{-j_L}|\tau|_\infty \right) \|x^{(0)}\|. \tag{13}$$

Theorem 2 gauges how approximately equivariant ScDCFNet is if the input undergoes not only a scale change $D_{\beta,v}$ but also a nonlinear spatial deformation D_τ, which is important both in theory and in practice because the scaling of an object is rarely perfect in reality. However, Theorem 2 only considers the stability of the representation under the ideal setting where layerwise feature maps $x^{(l)}(u, \alpha, \lambda)$ are computed (and stored) for all scales $\alpha \in \mathbb{R}$. For practical implementation (to be discussed in more detail in Section 5), the scale channel $\mathcal{S} \cong \mathbb{R}$ needs to be truncated to a finite interval $I = [-T, T] \subset \mathcal{S}$, i.e., $x^{(l)}(u, \alpha, \lambda)$ is only computed for $\alpha \in I$, which unavoidably destroys the global scaling symmetry—similar to the fact that truncating an image to a finite spatial support destroys translation symmetry. In practice, given the truncated l-th layer feature map $x^{(l-1)}(u, \alpha, \lambda)$ computed only for $\alpha \in I$, one typically first conducts a zero-padding to $x^{(l-1)}(u, \alpha, \lambda)$ in the scale channel before performing the convolution in scale (Worrall and Welling, 2019; Sosnovik et al., 2020). This, however, leads to a significant boundary "leakage" effect which destroys equivariance of the representation (see Section 6 for detailed empirical examination of such boundary effect.) We thus need to find a way to alleviate such boundary "leakage" issue and analyze its efficacy by studying the equivariance error (13) after a scale channel truncation.

The idea is very simple: after taking a closer look at the definition of the first layer operation (5), one can notice that, ignoring the bias $b^{(1)}(\lambda)$ and the nonlinear operator σ, it is essentially the convolution of $x^{(0)}$ with an L^1-normalized kernel $2^{-2\alpha} W^{(1)}_{\lambda',\lambda}(2^{-\alpha} u')$:

$$\begin{aligned} x^{(1)}[x^{(0)}](u, \alpha, \lambda) &= \sigma \left(\sum_{\lambda'} \int_{\mathbb{R}^2} 2^{-2\alpha} x^{(0)}(u + u', \lambda') W^{(1)}_{\lambda',\lambda} \left(2^{-\alpha} u'\right) du' + b^{(1)}(\lambda) \right) \\ &= \sigma \left(\sum_{\lambda'} x^{(0)}(\cdot, \lambda') * W^{(1)}_{\lambda',\lambda,\alpha}(u) + b^{(1)}(\lambda) \right), \end{aligned}$$

where $W^{(1)}_{\lambda',\lambda,\alpha}(u) \coloneqq 2^{-2\alpha} W^{(1)}_{\lambda',\lambda}(2^{-\alpha} u')$ forms a (yet to be normalized) mollifier, i.e., an approximation to identity, in $\mathbb{R}^2$ (Evans, 2010). Based on the basic properties of mollifiers (Evans, 2010), if $x^{(0)}(\cdot, \lambda')$ is a continuous function of u in $\mathbb{R}^2$, then

$$\lim_{\alpha \to -\infty} x^{(1)}[x^{(0)}](u, \alpha, \lambda) = \lim_{\alpha \to -\infty} \sigma \left(\sum_{\lambda'} x^{(0)}(\cdot, \lambda') * W^{(1)}_{\lambda',\lambda,\alpha}(u) + b^{(1)}(\lambda) \right) \tag{14}$$

$$= \sigma \left(\sum_{\lambda'} A_{\lambda',\lambda} x^{(0)}(u, \lambda') + b^{(1)}(\lambda) \right), \tag{15}$$

where $A_{\lambda',\lambda} = \int_{\mathbb{R}^2} W^{(1)}_{\lambda',\lambda}(u) du$. In particular, (15) implies that the limit of $x^{(1)}[x^{(0)}](u, \alpha, \lambda)$ exists as $\alpha \to -\infty$. This observation motivates us to consider the one-sided-replicate-padding in the scale channel (i.e., extending the truncated scale channel $I = [-T, T]$ beyond the

left end point $\alpha = -T$ according to Neumann boundary condition.) More specifically, the layerwise operations are now defined as follows: for the first layer,

$$\tilde{x}^{(1)}[x^{(0)}](u,\alpha,\lambda) = \begin{cases} x^{(1)}[x^{(0)}](u,\alpha,\lambda), & \alpha \in [-T,T], \\ x^{(1)}[x^{(0)}](u,-T,\lambda), & \alpha \in (-\infty,-T], \end{cases} \tag{16}$$

i.e., the computation remains the same as (5) for $\alpha \in I = [-T,T]$, and the value $x^{(1)}[x^{(0)}](u,\alpha,\lambda)$ at the left end point $\alpha = -T$ is extended beyond the truncated scale interval I for $\alpha \leq -T$ such that the next layer scale convolution can be computed on I. Similarly, the computations of the subsequent layer are

$$\tilde{x}^{(l)}[\tilde{x}^{(l-1)}](u,\alpha,\lambda) = \begin{cases} x^{(l)}[\tilde{x}^{(l-1)}](u,\alpha,\lambda), & \alpha \in [-T,T], \\ x^{(l)}[\tilde{x}^{(l-1)}](u,-T,\lambda), & \alpha \in (-\infty,-T]. \end{cases} \tag{17}$$

The following theorem quantifies the extra equivariance error incurred from scale channel truncation after the one-sided-replicate-padding (16) and (17) is adopted.

Theorem 3. *Under the same assumption of Theorem 2, if the layerwise operations are defined as* (16) *and* (17)*, given an input $x^{(0)}(\cdot,\lambda) \in H^1(\mathbb{R}^2)$ (the Sobolev space of functions on $\mathbb{R}^2$ with square-integrable first-order derivatives), we have, for any L,*

$$\left\| \tilde{x}^{(L)}[D_{\beta,v} \circ D_\tau x^{(0)}] - T_{\beta,v}\tilde{x}^{(L)}[x^{(0)}] \right\| \leq 2^{\beta+1} \left(4L|\nabla\tau|_\infty + 2^{-j_L}|\tau|_\infty\right) \|x^{(0)}\| + O(2^{-T}), \tag{18}$$

where the supremum over α in the definition of $\|\cdot\|$ on the left hand side of (18) *is taken for $\alpha \leq \min(T, T-\beta)$, the common scale domain of the feature maps $\tilde{x}^{(L)}[D_{\beta,v} \circ D_\tau x^{(0)}]$ and $T_{\beta,v}\tilde{x}^{(L)}[x^{(0)}]$, i.e.,*

$$\begin{aligned} &\left\| \tilde{x}^{(L)}[D_{\beta,v} \circ D_\tau x^{(0)}] - T_{\beta,v}\tilde{x}^{(L)}[x^{(0)}] \right\|^2 \\ &= \sup_{\alpha \leq \min(T,T-\beta)} \frac{1}{M_L} \sum_{\lambda=1}^{M_L} \int \left| \tilde{x}^{(L)}[D_{\beta,v} \circ D_\tau x^{(0)}] - T_{\beta,v}\tilde{x}^{(L)}[x^{(0)}] \right|^2 (u,\alpha,\lambda) du. \end{aligned}$$

On the contrary, if zero-padding in the scale channel is adopted (which is typically the case such as (Worrall and Welling, 2019; Sosnovik et al., 2020)) instead of the one-sided-replicate-padding (16) *and* (17)*, the scale channel truncation error in* (18) *will be $O(1)$ instead of $O(2^{-T})$.*

Theorem 3 demonstrates that the layerwise operations defined in (16) and (17) significantly alleviate the unavoidable boundary "leakage" effect incurred from scale channel truncation, contributing to the equivariance error (13) an exponentially decaying term in T, the length of the truncated scale interval. We will empirically demonstrate this theoretical result in Section 6.

5. Implementation

We discuss, in this section, the implementation details of ScDCFNet, including scale channel truncation and feature map discretization, basis and filter generation, discrete scale-space joint convolution, and batch-normalization.

5.1 Scale Channel Truncation And Feature Map Discretization

As mentioned in Section 4, in practice, the layerwise feature maps $x^{(l)}(u, \alpha, \lambda)$ can only be computed on a truncated scale interval $I = [-T, T] \subset \mathbb{R}$, which is discretized into a uniform grid of size N_s. Let H_l, W_l, M_l, respectively, be the height, width, and the number of unstructured channels of the l-th layer feature map $x^{(l)}$, then the input $x^{(0)}(u, \lambda)$ is stored as an array of shape $[M_0, H_0, W_0]$, and the layerwise output $x^{(l)}(u, \alpha, \lambda)$ has shape $[M_l, N_s, H_l, W_l]$.

Remark 5. *Due to, for example, interpolation and numerical integration, feature map discretization incurs an extra error to the equivariance of representation. Fortunately, the stability analysis in Theorem 2 under the continuous setting suggests that some of these errors can be mitigated. For instance, interpolating a rescaled digital image can be modeled as a perfect spatial rescaling followed by a small local distortion, the error induced by which in representation equivariance can be controlled in ScDCFNet thanks to Theorem 2.*

5.2 Basis And Filter Generation

Let K and K_α, respectively, be the numbers of the low-frequency components to be kept in the separable basis expansion in the spatial and scale domains. The spatial basis functions together with their rescaled versions $\{2^{-2\alpha}\psi_k(2^{-\alpha}u')\}_{k,\alpha,u'}$ are sampled on a uniform spatial grid of size $L \times L$ and stored as a tensor of shape $[K, N_s, L, L]$; the basis functions in scale $\{\varphi_m(\alpha')\}_{m,\alpha'}$ supported on the interval $I_\alpha \ni 0$ is sampled on a uniform grid of size L_α and stored as a tensor of shape $[K_\alpha, L_\alpha]$.

For the first layer, the truncated expansion coefficients $\{a^{(1)}_{\lambda',\lambda}(k)\}_{\lambda',\lambda,k}$ forming an array of shape $[M_0, M_1, K]$ are the trainable parameters of ScDCFNet at this layer, and the multiscale convolutional filters $\{2^{-2\alpha}W^{(1)}_{\lambda',\lambda}(2^{-\alpha}u')\}_{\lambda',\lambda,\alpha,u'}$ are the linear combinations of the spatial basis functions $\{2^{-2\alpha}\psi_k(2^{-\alpha}u')\}_{k,\alpha,u'}$ under the coefficients $\{a^{(1)}_{\lambda',\lambda}(k)\}_{\lambda',\lambda,k}$, stored as an array of shape $[M_0, M_1, N_s, L, L]$.

When $l > 1$, the tensor consisting of the l-th layer trainable coefficients $\{a^{(l)}_{\lambda',\lambda}(k, m)\}_{\lambda',\lambda,k,m}$ has shape $[M_{l-1}, M_l, K, K_\alpha]$. The joint convolutional filters $\{2^{-2\alpha}W^{(l)}_{\lambda',\lambda}(2^{-\alpha}u', \alpha')\}_{\lambda',\lambda,\alpha,\alpha',u'}$ of this layer are the linear combinations of the separable function bases $\{2^{-2\alpha}\psi_k(2^{-\alpha}u')\}_{k,\alpha,u'}$ and $\{\varphi_m(\alpha')\}_{m,\alpha'}$ under the coefficients $\{a^{(l)}_{\lambda',\lambda}(k, m)\}_{\lambda',\lambda,k,m}$, i.e., they are stored as an array of size $[M_{l-1}, M_l, N_s, L_\alpha, L, L]$ which is the tensor product of the first two arrays followed by a contraction with the third.

5.3 Discrete Scale-Space Joint Convolution

Given an input signal $x^{(0)}(u, \lambda)$ of shape $[M_0, H_0, W_0]$, a multiscale discrete spatial convolution, i.e., the discrete counterpart of (5) where integrals are replaced by summations, with the filter $\{2^{-2\alpha}W^{(1)}_{\lambda',\lambda,\alpha,u'}\}_{\lambda',\lambda,\alpha,u'}$ of shape $[M_0, M_1, N_s, L, L]$ is conducted to obtain the first layer feature $x^{(1)}(u, \alpha, \lambda)$ of shape $[M_1, N_s, H_1, W_1]$. More specifically, the standard 2D convolution is performed on the input with the filter reshaped to $[M_0, M_1N_s, L, L]$, producing an output of shape $[M_1N_s, H_1, W_1]$, which is subsequently resized to $[M_1, N_s, H_1, W_1]$.

Starting from the second layer, let $x^{(l-1)}(u, \alpha, \lambda)$ be the feature of shape $[M_{l-1}, N_s, H_{l-1}, W_{l-1}]$, and $F = \{2^{-2\alpha} W^{(l)}_{\lambda',\lambda}(2^{-\alpha}u', \alpha')\}_{\lambda',\lambda,\alpha,\alpha',u'}$ be the filter of size $[M_{l-1}, M_l, N_s, L_\alpha, L, L]$. The signal $x^{(l-1)}$ is first shifted in the scale channel by $l_\alpha \in [0, L_\alpha - 1]$ according to one-sided-replicate-padding (c.f. Section 4), and then convolved with the filter $F[:, :, :, l_\alpha, :, :]$ (after reshaping) to obtain an output tensor of size $[M_l, N_s, H_l, W_l]$. We iterate over all $l_\alpha \in [0, L_\alpha - 1]$, and the l-th layer feature $x^{(l)}(u, \alpha, \lambda)$ is the sum of the L_α tensors.

For computer vision tasks where scale-invariant features are preferred, e.g., image classification, a max-pooling in the scale channel is conducted on the last layer feature $x^{(L)}(u, \alpha, \lambda)$ of size $[M_L, N_s, H_L, W_L]$, producing a scale-invariant output of shape $[M_L, H_L, W_L]$. Without explicitly mentioning, such max-pooling in scale is only performed in the last layer.

5.4 Batch-Normalization

Batch-normalization (Ioffe and Szegedy, 2015) accelerates network training by reducing layerwise covariate shift, and it has become an integral part in various CNN architectures. With an extra scale index α in the feature map $x^{(l)}(u, \alpha, \lambda)$ of an ScDCFNet, we need to include α in the normalization in order not to destroy $\mathcal{ST}$-equivariance, i.e., a batch of features $\{x^{(l)}_n(u, \alpha, \lambda)\}_{n=1}^N$ should be normalized as if it were a collection of 3D data (two dimensions for u, and one dimension for α.)

6. Numerical Experiments

In this section, we conduct several numerical experiments for the following three purposes.

1. To demonstrate that ScDCFNet robustly achieves $\mathcal{ST}$-equivariance (4) in the practical setting where signals are discrete and scale channels are truncated, verifying the theoretical results Theorem 2 and Theorem 3.

2. To illustrate that ScDCFNet significantly outperforms regular CNNs as well as other competing scale-invariant/equivariant networks at a much reduced model size in multi-scale image classification.

3. To show that a trained ScDCFNet auto-encoder is able to reconstruct rescaled versions of the input by simply applying group actions on the image codes, demonstrating that ScDCFNet indeed explicitly encodes the input scale information into the representation.

6.1 Data Sets And Models

The experiments are conducted on the Scaled MNIST (SMNIST), Scaled Fashion-MNIST (SFashion), and the STL-10 data sets (Coates et al., 2011).

SMNIST and SFashion are built by rescaling the original MNIST (LeCun et al., 1998) and Fashion-MNIST (Xiao et al., 2017) images by a factor randomly sampled from a uniform distribution on $[0.3, 1]$. A zero-padding back to a size of 28×28 is conducted after the rescaling. If mentioned explicitly, for some experiments, the images are resized to 56×56 for better visualization.

The STL-10 data set is comprised of 5,000 training and 8,000 testing labeled RGB images belonging to 10 classes such as airplane, bird, and car. The images have a spatial resolution

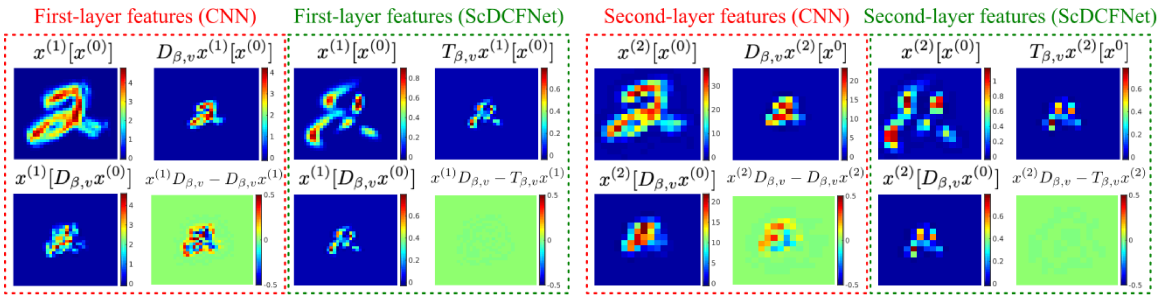


Figure 2: Verification of $\mathcal{ST}$-equivariance in Section 6.2. Given the original input $x^{(0)}$ and its rescaled version $D_{\beta,v}x^{(0)}$, the four figures in each dashed rectangle are: $x^{(l)}[x^{(0)}]$ (l-th layer feature of the original input), $x^{(l)}[D_{\beta,v}x^{(0)}]$ (l-th layer feature of the rescaled input), $T_{\beta,v}x^{(l)}[x^{(0)}]$ (rescaled l-th layer feature of the original input), and the difference $(x^{(l)}[D_{\beta,v}x^{(0)}] - T_{\beta,v}x^{(l)}[x^{(0)}])$ displayed in a (signal intensity) scale relative to the maximum value of $x^{(l)}[D_{\beta,v}x^{(0)}]$. It is clear that even after numerical discretization, $\mathcal{ST}$-equivariance still approximately holds for ScDCFNet, i.e., $x^{(l)}[D_{\beta,v}x^{(0)}] - T_{\beta,v}x^{(l)}[x^{(0)}] \approx 0$, but not for a regular CNN.

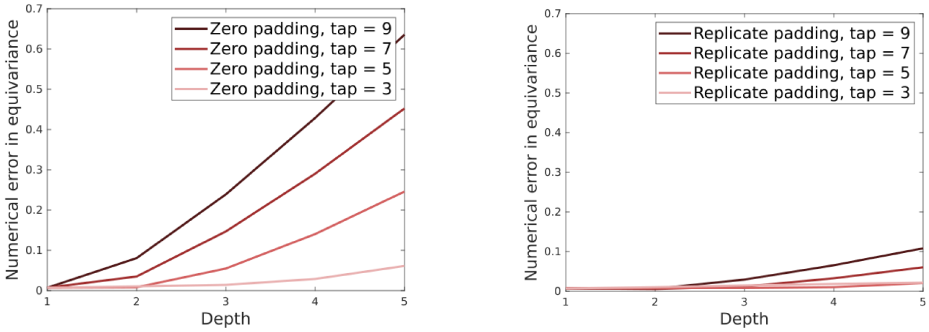
of 96×96, and we do not rescale the images due to the rich scaling variance already existing in the data set.

We compare the performance of our ScDCFNet with other network models that deal with scaling variation within the input data, including the local scale-invariant models LSI-CNN (Kanazawa et al., 2014), SS-CNN (Ghosh and Gupta, 2019), and scale-equivariant models SI-CNN (Xu et al., 2014), SEVF (Marcos et al., 2018), DSS (Worrall and Welling, 2019), and SESN (Sosnovik et al., 2020).

6.2 Verification of $\mathcal{ST}$-Equivariance

We first verify that ScDCFNet indeed achieves $\mathcal{ST}$-equivariance (4). Specifically, we compare the feature maps of a two-layer ScDCFNet with randomly generated truncated filter expansion coefficients and those of a regular CNN. The exact architectures are detailed in Appendix B.1. Figure 2 displays the first- and second-layer feature maps of an original image $x^{(0)}$ and its rescaled version $D_{\beta,v}x^{(0)}$ using the two comparing architectures. Feature maps at different layers are rescaled to the same spatial dimension for visualization. The four images enclosed in each of the dashed rectangle correspond to: $x^{(l)}[x^{(0)}]$ (l-th layer feature of the original input), $x^{(l)}[D_{\beta,v}x^{(0)}]$ (l-th layer feature of the rescaled input), $T_{\beta,v}x^{(l)}[x^{(0)}]$ (rescaled l-th layer feature of the original input, where $T_{\beta,v}$ is understood as $D_{\beta,v}$ for a regular CNN due to the lack of a scale index α), and the difference $x^{(l)}[D_{\beta,v}x^{(0)}] - T_{\beta,v}x^{(l)}[x^{(0)}]$. It is clear that even with numerical discretization, which can be modeled as a form of input deformation, ScDCFNet is still approximately $\mathcal{ST}$-equivariant, i.e., $x^{(l)}[D_{\beta,v}x^{(0)}] \approx T_{\beta,v}x^{(l)}[x^{(0)}]$, whereas a regular CNN does not have such a property.

We also examine how the numerical error in equivariance incurred by the boundary "leakage" effect after the scale channel truncation (cf. Section 4) evolves as the network gets deeper. In particular, we aim to empirically verify the efficacy of the one-sided-replicate-padding for scale channel convolution in alleviating the "leakage" effect; see Theorem 3. In comparison, we note that another effective way explored by Worrall and Welling (2019)



(a) Zero-padding for the scale channel.

(b) One-sided-replicate-padding for the scale channel.

Figure 3: The numerical error in equivariance incurred by the boundary "leakage" effect after scale channel truncation as a function of network depth. Either (a) zero-padding or (b) one-sided-replicate-padding is used for the convolution in scale. The error is unavoidable as depth becomes larger, but it can be mitigated by (1) using joint convolutional filters with a smaller support I_α in scale (i.e., a smaller number of "taps" L_α after discretization), and (2) using a one-sided-replicate-padding instead of zero-padding.

and Sosnovik et al. (2020) to reduce the boundary "leakage" phenomenon is to choose a much smaller support I_α for the convolutional filters $W^{(l)}_{\lambda',\lambda}(u',\alpha')$ in scale compared to the truncated scale interval I, i.e., $|I_\alpha| \ll |I|$; equivalently, this means the filter size L_α in scale, or the number of "taps", is much smaller compared to the number of scale channels N_s in the feature map. The error in equivariance is measured in a relative L^2 sense at a particular scale α, i.e.,

$$\text{Error} = \|x^{(l)}[D_{\beta,v}x^{(0)}](\cdot,\alpha) - T_{\beta,v}x^{(l)}[x^{(0)}](\cdot,\alpha)\|_{L^2} / \|T_{\beta,v}x^{(l)}[x^{(0)}](\cdot,\alpha)\|_{L^2}.$$

It is clear from Figure 3 that the boundary "leakage" effect is unavoidable as the network becomes deeper. However, the error can be significantly reduced by either choosing joint convolutional filters with a smaller support I_α in scale, or using a one-sided-replicate-padding instead of the default zero-padding typically adopted in other works (Worrall and Welling, 2019; Sosnovik et al., 2020). Using the correct padding has a stronger effect in reducing the equivariance error than having a smaller support I_α, and combining the two practices leads to a decrease in equivariance error by more than an order of magnitude in deep networks.

6.3 Multiscale Image Classification

We next demonstrate the improved performance of ScDCFNet in multiscale image classification compared to regular CNNs and other scale-invariant or scale-equivariant models.

6.3.1 SMNIST And SFashion

We first test the comparing models on the SMNIST and SFashion data sets. Five independent realizations of the rescaled data sets are generated according to Section 6.1 and kept the same for all networks. Each rescaled data set is split into 10,000 images for training, 2,000 images for evaluation, and 50,000 images for testing. In order to demonstrate the performance of

	SMNIST (28×28) test accuracy (%)			SMNIST (28×28)+ test accuracy (%)		
Models	$N_{\text{tr}} = 2,000$	$N_{\text{tr}} = 5,000$	$N_{\text{tr}} = 10,000$	$N_{\text{tr}} = 2,000$	$N_{\text{tr}} = 5,000$	$N_{\text{tr}} = 10,000$
CNN	95.01 ± 0.27	96.39 ± 0.12	97.41 ± 0.13	96.36 ± 0.43	97.27 ± 0.14	98.05 ± 0.07
SS-CNN	95.43 ± 0.21	96.70 ± 0.20	97.64 ± 0.08	96.13 ± 0.25	96.84 ± 0.17	97.98 ± 0.03
SEVF	95.09 ± 0.15	96.29 ± 0.15	97.28 ± 0.16	96.06 ± 0.25	96.68 ± 0.09	97.74 ± 0.14
LSI-CNN	95.46 ± 0.22	96.64 ± 0.08	97.56 ± 0.13	96.43 ± 0.23	97.17 ± 0.08	97.97 ± 0.05
SI-CNN	95.17 ± 0.21	96.58 ± 0.22	97.53 ± 0.12	96.44 ± 0.09	97.21 ± 0.27	98.08 ± 0.09
DSS	95.06 ± 0.24	96.42 ± 0.17	97.34 ± 0.13	96.56 ± 0.13	97.24 ± 0.10	98.03 ± 0.06
SESN	95.75 ± 0.21	96.87 ± 0.12	97.81 ± 0.13	96.61 ± 0.14	97.30 ± 0.15	98.11 ± 0.09
ScDCFNet	$\mathbf{95.87 \pm 0.18}$	$\mathbf{97.09 \pm 0.05}$	$\mathbf{97.91 \pm 0.08}$	$\mathbf{96.68 \pm 0.13}$	$\mathbf{97.40 \pm 0.18}$	$\mathbf{98.19 \pm 0.07}$

	SMNIST (56×56) test accuracy (%)			SMNIST (56×56)+ test accuracy (%)		
Models	$N_{\text{tr}} = 2,000$	$N_{\text{tr}} = 5,000$	$N_{\text{tr}} = 10,000$	$N_{\text{tr}} = 2,000$	$N_{\text{tr}} = 5,000$	$N_{\text{tr}} = 10,000$
CNN	96.15 ± 0.19	97.20 ± 0.10	98.01 ± 0.08	97.17 ± 0.12	97.75 ± 0.12	98.32 ± 0.04
SS-CNN	96.32 ± 0.12	97.37 ± 0.22	98.08 ± 0.12	96.90 ± 0.18	97.57 ± 0.09	98.28 ± 0.14
SEVF	96.23 ± 0.17	97.19 ± 0.13	98.01 ± 0.12	96.86 ± 0.14	97.51 ± 0.08	98.20 ± 0.20
LSI-CNN	96.43 ± 0.18	97.34 ± 0.17	98.13 ± 0.13	97.11 ± 0.16	97.66 ± 0.07	98.29 ± 0.16
SI-CNN	95.97 ± 0.18	97.24 ± 0.12	97.98 ± 0.14	97.16 ± 0.12	97.78 ± 0.06	98.44 ± 0.08
DSS	96.10 ± 0.18	97.26 ± 0.10	97.97 ± 0.10	97.19 ± 0.08	97.83 ± 0.10	98.41 ± 0.11
SESN	96.50 ± 0.19	97.52 ± 0.10	98.19 ± 0.12	97.21 ± 0.20	97.80 ± 0.11	98.40 ± 0.09
ScDCFNet	$\mathbf{96.75 \pm 0.18}$	$\mathbf{97.61 \pm 0.11}$	$\mathbf{98.27 \pm 0.09}$	$\mathbf{97.32 \pm 0.08}$	$\mathbf{97.85 \pm 0.15}$	$\mathbf{98.46 \pm 0.09}$

Table 1: Classification accuracy on the SMNIST data set. Models are trained on $N_{\text{tr}} = 2$K, 5K, or 10K images with spatial resolution 28×28 or 56×56. A plus sign "+" is used to denote the presence of scaling data augmentation during training. Test accuracies are reported as mean $\pm$ std over five independent realizations of the rescaled data set.

various models in the small data regime, we also report the results trained with 2,000 and 5,000 images.

Following the experimental setup by Ghosh and Gupta (2019), a baseline CNN consisting of three convolutional and two fully-connected layers with batch-normalization is used as a benchmark. The number of output channels of the three convolutional and the first fully-connected layers are set to $[32, 63, 95, 256]$. Convolutional filters of size 7×7 are used in each layer. All comparing networks are built on the same CNN baseline, and the number of trainable parameters are kept almost the same across different models by varying the number of unstructured channels. For equivariant models, a max-pooling in the scale channel is performed only after the final convolutional layer to produce scale-invariant features for classification. For $\mathcal{ST}$-equivariant models achieved by space-scale joint convolution such as SESN and ScDCFNet, the filter size in scale L_α is set to 3. Finally, for ScDCFNet, we set $K = 15$, $K_\alpha = 3$, and $N_s = 5$.

All networks are trained with the Adam optimizer (Kingma and Ba, 2014) for 60 epochs with a batch size of 128. The initial learning rate is set to 0.01 and scheduled to decrease tenfold after 20 and 40 epochs. We conduct the experiments in 12 different settings, where

- the input images size is either 28×28 or 56×56;
- the models are trained with or without scaling data augmentation;
- the number of training samples N_{tr} is 2,000, 5,000, or 10,000.

We conduct the experiments on five independent realizations of the rescaled data, and report the mean $\pm$ std of the test accuracy in Table 1 and Table 2, where, for instance, (28×28) (or (28×28)+) denotes models are trained on images of size 28×28 without (or with)

	SFashion (28×28) test accuracy (%)			SFashion (28×28)+ test accuracy (%)		
Models	$N_{\text{tr}} = 2,000$	$N_{\text{tr}} = 5,000$	$N_{\text{tr}} = 10,000$	$N_{\text{tr}} = 2,000$	$N_{\text{tr}} = 5,000$	$N_{\text{tr}} = 10,000$
CNN	79.88 ± 0.60	82.71 ± 0.19	84.60 ± 0.30	81.82 ± 0.49	83.85 ± 0.51	86.53 ± 0.14
SS-CNN	79.24 ± 0.18	82.83 ± 0.49	85.39 ± 0.32	80.15 ± 0.31	83.09 ± 0.14	85.89 ± 0.19
SEVF	79.01 ± 0.45	81.76 ± 0.40	84.73 ± 0.11	79.47 ± 0.44	82.36 ± 0.30	85.23 ± 0.16
LSI-CNN	79.20 ± 0.78	82.58 ± 0.52	85.16 ± 0.14	80.15 ± 0.71	83.10 ± 0.26	86.06 ± 0.16
SI-CNN	79.95 ± 0.53	83.36 ± 0.29	85.32 ± 0.22	82.27 ± 0.46	84.09 ± 0.40	86.85 ± 0.15
DSS	79.82 ± 0.44	82.90 ± 0.22	84.50 ± 0.51	82.20 ± 0.51	84.04 ± 0.24	86.51 ± 0.27
SESN	80.88 ± 0.51	83.78 ± 0.27	85.93 ± 0.28	82.34 ± 0.52	84.26 ± 0.23	86.90 ± 0.27
ScDCFNet	$\mathbf{81.32 \pm 0.41}$	$\mathbf{84.24 \pm 0.35}$	$\mathbf{86.19 \pm 0.15}$	$\mathbf{82.42 \pm 0.38}$	$\mathbf{84.31 \pm 0.30}$	$\mathbf{87.10 \pm 0.34}$
	SFashion (56×56) test accuracy (%)			SFashion (56×56)+ test accuracy (%)		
Models	$N_{\text{tr}} = 2,000$	$N_{\text{tr}} = 5,000$	$N_{\text{tr}} = 10,000$	$N_{\text{tr}} = 2,000$	$N_{\text{tr}} = 5,000$	$N_{\text{tr}} = 10,000$
CNN	81.17 ± 0.39	84.05 ± 0.12	85.84 ± 0.47	83.35 ± 0.16	85.16 ± 0.22	87.54 ± 0.18
SS-CNN	80.97 ± 0.30	83.95 ± 0.33	86.42 ± 0.23	81.48 ± 0.75	84.24 ± 0.38	86.69 ± 0.23
SEVF	81.22 ± 0.35	84.13 ± 0.27	86.37 ± 0.30	81.88 ± 0.47	84.28 ± 0.32	86.96 ± 0.17
LSI-CNN	81.78 ± 0.47	84.49 ± 0.30	86.94 ± 0.34	82.44 ± 0.53	84.71 ± 0.26	87.32 ± 0.16
SI-CNN	81.11 ± 0.48	84.18 ± 0.34	86.22 ± 0.17	83.55 ± 0.40	85.18 ± 0.49	87.97 ± 0.16
DSS	81.41 ± 0.32	84.15 ± 0.25	85.91 ± 0.23	83.54 ± 0.40	85.44 ± 0.40	87.63 ± 0.07
SESN	82.34 ± 0.51	85.03 ± 0.14	86.95 ± 0.23	83.61 ± 0.40	85.40 ± 0.23	87.91 ± 0.13
ScDCFNet	$\mathbf{83.01 \pm 0.31}$	$\mathbf{85.79 \pm 0.26}$	$\mathbf{87.63 \pm 0.15}$	$\mathbf{84.00 \pm 0.41}$	$\mathbf{86.14 \pm 0.25}$	$\mathbf{88.18 \pm 0.17}$

Table 2: Classification accuracy on the SFashion data set. Models are trained on $N_{\text{tr}} = 2$K, 5K, or 10K images with spatial resolution 28×28 or 56×56. A plus sign "+" is used to denote the presence of scaling data augmentation during training. Test accuracies are reported as mean $\pm$ std over five independent realizations of the rescaled data set.

data augmentation. It can be observed from Table 1 and Table 2 that ScDCFNet consistently outperforms all comparing methods in every experimental setting, and the improvement in accuracy is especially pronounced in the small data regime without data augmentation.

We want to note that the results of SESN displayed in Table 1 is slightly worse than those reported by Sosnovik et al. (2020)—this is because even though Sosnovik et al. (2020) also propose space-scale joint convolution to achieve $\mathcal{ST}$-equivariance, it is however not implemented in their work for the SMNIST experiment, i.e., they set the support of the convolutional filters in scale L_α to 1 instead of 3 (or no "interscale interaction" according to the terminology by Sosnovik et al. (2020).) The deterioration in test performance of SESN when $L_\alpha > 1$ is in line with the finding by Sosnovik et al. (2020) on the STL-10 data set that "interscale interaction" significantly reduces the accuracy of SESN due to the high equivariance error introduced by the boundary "leakage" effect. This is however successfully mitigated in ScDCFNet, verifying again our theoretical result Theorem 3.

We next conduct an ablation study on the role of K and K_α, i.e., the numbers of the truncated basis functions in space and scale, on the performance of ScDCFNet. The results of ScDCFNet on the SMNIST and SFashion data sets with varying K and K_α are shown in Table 3. It can be observed that enlarging K and K_α, which increases the expressive power of the network at the cost of more trainable parameters (i.e., larger model size), indeed boosts the performance of ScDCFNet. However, the accuracy gradually plateaus around $K \approx 12$ and $K_\alpha \approx 3$. This is because high frequency filters suffer from aliasing effect after discretization, thus introducing a larger equivariance error. This phenomenon will be further explore in Section 6.4.

ScDCFNet			SMNIST (28×28) accuracy (%)		SMNIST (28×28)+ accuracy (%)	
K_α	K	# Params	$N_{\text{tr}} = 2,000$	$N_{\text{tr}} = 5,000$	$N_{\text{tr}} = 2,000$	$N_{\text{tr}} = 5,000$
3	15	1.00	95.87 ± 0.18	$\mathbf{97.09 \pm 0.05}$	$\mathbf{96.68 \pm 0.13}$	$\mathbf{97.40 \pm 0.18}$
3	12	0.80	$\mathbf{95.88 \pm 0.21}$	97.06 ± 0.07	96.67 ± 0.19	97.38 ± 0.16
3	10	0.67	95.84 ± 0.18	97.00 ± 0.08	96.49 ± 0.09	97.23 ± 0.12
3	6	0.40	95.61 ± 0.18	96.89 ± 0.19	96.50 ± 0.15	97.17 ± 0.15
2	15	0.67	95.77 ± 0.15	97.04 ± 0.05	$\mathbf{96.68 \pm 0.31}$	97.34 ± 0.08
2	10	0.44	95.72 ± 0.23	96.97 ± 0.07	96.52 ± 0.11	97.30 ± 0.13
2	6	0.27	95.68 ± 0.17	96.92 ± 0.13	96.50 ± 0.14	97.09 ± 0.11
1	15	0.33	95.84 ± 0.11	96.94 ± 0.12	96.65 ± 0.19	97.38 ± 0.07
1	10	0.22	95.62 ± 0.22	96.93 ± 0.11	96.61 ± 0.12	97.26 ± 0.21
1	6	0.13	95.74 ± 0.17	96.82 ± 0.06	96.48 ± 0.13	97.17 ± 0.16
ScDCFNet			SFashion (28×28) accuracy (%)		SFashion (28×28)+ accuracy (%)	
K_α	K	# Params	$N_{\text{tr}} = 2,000$	$N_{\text{tr}} = 5,000$	$N_{\text{tr}} = 2,000$	$N_{\text{tr}} = 5,000$
3	15	1.00	$\mathbf{81.32 \pm 0.41}$	$\mathbf{84.24 \pm 0.35}$	$\mathbf{82.42 \pm 0.38}$	84.31 ± 0.30
3	12	0.80	81.23 ± 0.37	84.17 ± 0.30	82.39 ± 0.46	84.22 ± 0.40
3	10	0.67	81.03 ± 0.38	83.75 ± 0.23	81.97 ± 0.50	83.96 ± 0.42
3	6	0.40	80.88 ± 0.27	83.83 ± 0.11	81.03 ± 0.22	83.66 ± 0.21
2	15	0.67	80.88 ± 0.31	84.05 ± 0.40	82.18 ± 0.34	$\mathbf{84.40 \pm 0.30}$
2	10	0.44	80.65 ± 0.50	83.89 ± 0.23	81.86 ± 0.39	83.85 ± 0.39
1	6	0.27	80.82 ± 0.42	83.44 ± 0.29	81.46 ± 0.45	83.62 ± 0.24
1	15	0.33	80.74 ± 0.39	84.09 ± 0.27	82.33 ± 0.44	84.17 ± 0.32
1	10	0.22	80.89 ± 0.33	83.65 ± 0.44	81.87 ± 0.58	83.66 ± 0.23
1	6	0.13	80.60 ± 0.27	83.61 ± 0.33	81.40 ± 0.56	83.57 ± 0.48

Table 3: Ablation study on the role of K and K_α in the performance of ScDCFNet. The column "# Params" stands for the number of parameters of the current model compared to that of the benchmark ScDCFNet used in Table 1 and Table 2, i.e., $K = 15$ and $K_\alpha = 3$.

Models	Accuracy (%)
ResNet-16	81.98 ± 0.23
LSI-CNN	81.79 ± 0.53
SI-CNN	81.75 ± 0.14
SS-CNN	69.51 ± 1.27
DSS	82.21 ± 0.47
SESN	83.89 ± 0.05
ScDCFNet	$\mathbf{84.90 \pm 0.36}$

Table 4: Test accuracy on the STL-10 data set.

6.3.2 STL-10

We next conduct the experiments on the STL-10 data set to examine the performance of various models on natural image classification. We use a ResNet by He et al. (2016) with 16 layers as the baseline, upon which all models are built while the number of trainable parameters is kept almost the same. Similarly, a max-pooling in the scale channel is performed only after the final residual block for equivariant models. For SESN and ScDCFNet, the filter size in scale L_α is set to 2.

Following the idea of Sosnovik et al. (2020), we augment the data set during training by applying 12 pixel zero-padding followed by random cropping. In addition, images are randomly flipped horizontally and Cutout by DeVries and Taylor (2017) with 1 hole of 32 pixels is used. All models are trained for 1000 epochs with a batch size of 128. We use an

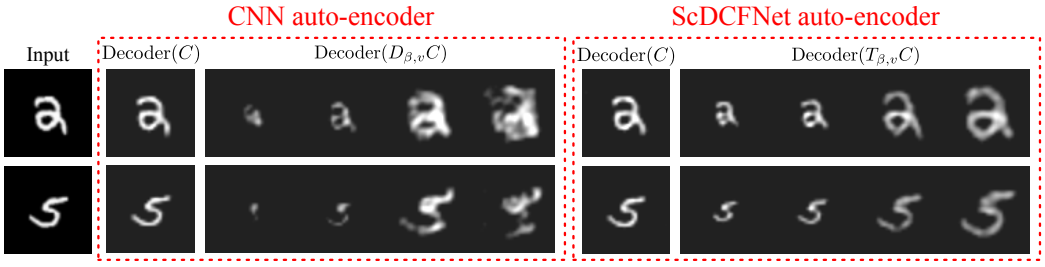


Figure 4: Reconstructing rescaled versions of the original test image by manipulating its image code C according to the group action (3). The first two images on the left are the original inputs; Decoder(C) denotes the reconstruction using the (unchanged) image code C; Decoder($D_{\beta,v}C$) and Decoder($T_{\beta,v}C$) denote the reconstructions using the "rescaled" image codes $D_{\beta,v}C$ and $T_{\beta,v}C$ respectively according to (2) and (3). Unlike the regular CNN auto-encoder, the ScDCFNet auto-encoder manages to generate rescaled versions of the original input, suggesting that it successfully encodes the scale information directly into the representation.

SGD optimizer with Nesterov momentum set to 0.9 and weight decay being 5×10^{-4}. The initial learning rate is set to 0.1 and scheduled to decrease tenfold after 300, 400, 600 and 800 epochs.

We run three independent trials of the experiment, and report the mean $\pm$ std of the test accuracy in Table 4. It is clear that ScDCFNet again achieves the best performance compared to other models, further demonstrating its advantage in multiscale image classification.

6.4 Image Reconstruction

In the last experiment, we illustrate the ability of ScDCFNet to explicitly encode the input scale information into the representation. To achieve this, we train an ScDCFNet auto-encoder on the SMNIST data set with images resized to 56×56 for better visualization. The encoder stacks two $\mathcal{ST}$-equivariant convolutional blocks with 2×2 average-pooling, and the decoder contains a succession of two transposed convolutional blocks with 2×2 upsampling. A regular CNN auto-encoder is also trained for comparison (see Table 5 in Appendix B.2 for the detailed architecture.)

Our goal is to demonstrate that the image code produced by the ScDCFNet auto-encoder contains the scale information of the input, i.e., by applying the group action $T_{\beta,v}$ (3) to the code C of a test image before feeding it to the decoder, we can reconstruct rescaled versions of original input. This property can be visually verified in Figure 4. In contrast, a regular CNN auto-encoder fails to do so.

Finally, we quantitatively study the impact of basis truncation in ScDCFNet on the reconstruction error. In the experiment, we fix the number of basis functions in scale at $K_\alpha = 3$, and train ScDCFNet auto-encoders with spatial basis functions truncated to $K = 5, 8, 12$, and 15. After training the networks, we calculate the relative L^2 distance between the rescaled original images $D_\beta[x^{(0)}]$ and their reconstructions at 13 different scales $2^\beta, \beta \in \{-0.6, -0.5, \cdots, 0.6\}$:

$$\text{Error} = \frac{\left\| \text{Decoder}\left\{T_\beta\left[\text{Encoder}\left(x^{(0)}\right)\right]\right\} - D_\beta\left[x^{(0)}\right] \right\|_{L^2}}{\left\| D_\beta\left[x^{(0)}\right] \right\|_{L^2}}. \tag{19}$$

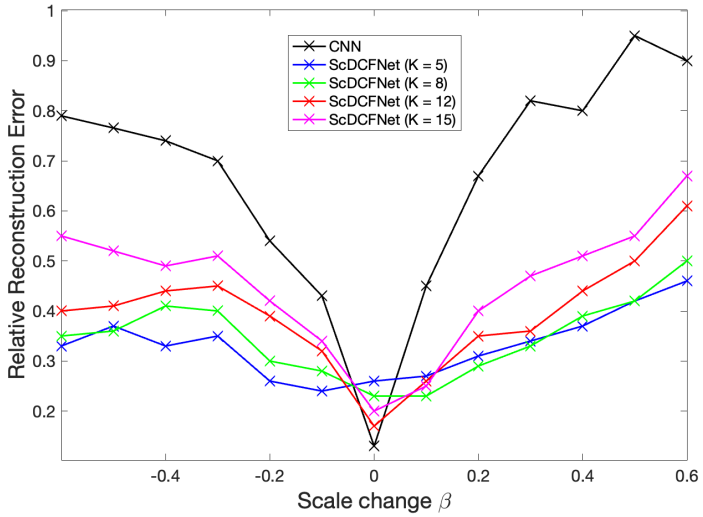


Figure 5: The relative error (19) between the rescaled inputs and their reconstructions using CNN and ScDCFNet autoencoders at 13 different scales $2^{\beta}, \beta \in \{-0.6, -0.5, \cdots, 0.6\}$. The means of the reconstruction error over the test images of SMNIST are displayed.

To adjust for the contrast loss visible in Figure 4, images are thresholded before calculating the error (19). We display the mean of the reconstruction error over the test set of SMNIST across 13 different scales in Figure 5. It can be observed that ScDCFNet autoencoders significantly outperform their CNN counterpart by having a smaller error when $\beta \neq 0$, i.e., reconstructing more accurately at a scale *different* from the original image. Moreover, by reducing the number of basis functions K, the reconstruction error becomes smaller, demonstrating again our theoretical results Theorem 2 and Theorem 3 that basis truncation improves deformation robustness of the equivariant representation.

7. Conclusion

We propose, in this paper, a $\mathcal{ST}$-equivariant CNN with joint convolutions across the space $\mathbb{R}^2$ and the scaling group $\mathcal{S}$, which we show to be both sufficient and necessary to achieve equivariance for the regular representation of the scaling-translation group. To reduce the computational cost and model complexity incurred by the joint convolutions, the convolutional filters supported on $\mathbb{R}^2 \times \mathcal{S}$ are decomposed under a separable basis across the two domains and truncated to only low-frequency components. Moreover, the truncated filter expansion leads also to improved deformation robustness of the equivariant representation, i.e., the model is still approximately equivariant even if the scaling transformation is imperfect. Experimental results suggest that ScDCFNet achieves improved performance in multiscale image classification with greater interpretability and reduced model size compared to regular CNN models.

For future work, we will study the application of ScDCFNet in other more complicated vision tasks, such as object detection/localization and pose estimation, where it is beneficial to directly encode the input scale information into the deep representation. We will explore

other efficient implementation of the model, e.g., using filter-bank type of techniques to compute convolutions with multiscale spatial filters, to further reduce the computational cost.

Acknowledgments

The research of WZ is partially supported by NSF under DMS-2052525 and DMS-2140982. QQ is partially supported by NSF DMS-1737744. The research of RC is supported in part by the Air Force Office of Scientific Research through award FA 9550-20-1-0266. GS is partially supported by NSF, NGA, and ONR. XC is partially supported by NSF (DMS-1820827) and the Alfred P. Sloan Foundation.

Appendix A. Proofs

In this appendix we provide proofs of the results that were stated without proof in the main text.

A.1 Proof of Theorem 1

Proof of Theorem 1. We note first that (4) holds true if and only if the following being valid for all $l \geq 1$,

$$T_{\beta,v} x^{(l)}[x^{(l-1)}] = x^{(l)}[T_{\beta,v} x^{(l-1)}], \tag{20}$$

where $T_{\beta,v} x^{(0)}$ is understood as $D_{\beta,v} x^{(0)}$. We also note that the layerwise operations of a general feedforward neural network with an extra index $\alpha \in \mathcal{S}$ can be written as

$$x^{(1)}[x^{(0)}](u,\alpha,\lambda) = \sigma\left(\sum_{\lambda'} \int_{\mathbb{R}^2} x^{(0)}(u+u',\lambda') W^{(1)}(u',\lambda',u,\alpha,\lambda) du' + b^{(1)}(\lambda)\right), \tag{21}$$

and, for $l > 1$,

$$\begin{aligned} x^{(l)}[x^{(l-1)}](u,\alpha,\lambda) =& \sigma\left(\sum_{\lambda'} \int_{\mathbb{R}^2} \int_{\mathbb{R}} x^{(l-1)}(u+u',\alpha+\alpha',\lambda') \right. \\ & \left. W^{(l)}(u',\alpha',\lambda',u,\alpha,\lambda) d\alpha' du' + b^{(l)}(\lambda)\right). \end{aligned} \tag{22}$$

To prove the sufficient part: when $l = 1$, (2), (3), and (5) lead to

$$\begin{aligned} & T_{\beta,v} x^{(1)}[x^{(0)}](u,\alpha,\lambda) = x^{(1)}[x^{(0)}]\left(2^{-\beta}(u-v), \alpha-\beta, \lambda\right) \\ =& \sigma\left(\sum_{\lambda'} \int x^{(0)}\left(2^{-\beta}(u-v)+u', \lambda'\right) W^{(1)}_{\lambda',\lambda}\left(2^{-(\alpha-\beta)}u'\right) 2^{-2(\alpha-\beta)} du' + b^{(1)}(\lambda)\right) \\ =& \sigma\left(\sum_{\lambda'} \int x^{(0)}\left(2^{-\beta}(u-v+\tilde{u}), \lambda'\right) W^{(1)}_{\lambda',\lambda}\left(2^{-\alpha}\tilde{u}\right) 2^{-2\alpha} d\tilde{u} + b^{(1)}(\lambda)\right), \end{aligned}$$

and

$$\begin{aligned}
&x^{(1)}[D_{\beta,v}x^{(0)}](u,\alpha,\lambda)\\
=&\sigma\left(\sum_{\lambda'}\int_{\mathbb{R}^2} D_{\beta,v}x^{(0)}(u+u',\lambda')W^{(1)}_{\lambda',\lambda}\left(2^{-\alpha}u'\right)2^{-2\alpha}du'+b^{(1)}(\lambda)\right)\\
=&\sigma\left(\sum_{\lambda'}\int_{\mathbb{R}^2} x^{(0)}(2^{-\beta}\left(u+u'-v\right),\lambda')\,W^{(1)}_{\lambda',\lambda}\left(2^{-\alpha}u'\right)2^{-2\alpha}du'+b^{(1)}(\lambda)\right).
\end{aligned}$$

Hence $T_{\beta,v}x^{(1)}[x^{(0)}]=x^{(1)}[D_{\beta,v}x^{(0)}]$.

When $l>1$, we have

$$\begin{aligned}
&T_{\beta,v}x^{(l)}[x^{(l-1)}](u,\alpha,\lambda)=x^{(l)}[x^{(l-1)}]\left(2^{-\beta}(u-v),\alpha-\beta,\lambda\right)\\
=&\sigma\left(\sum_{\lambda'}\int\int x^{(l-1)}\left(2^{-\beta}(u-v)+u',\alpha-\beta+\alpha',\lambda'\right)\cdot\right.\\
&\qquad\left.W^{(l)}_{\lambda',\lambda}\left(2^{-(\alpha-\beta)}u',\alpha'\right)2^{-2(\alpha-\beta)}du'd\alpha'+b^{(l)}(\lambda)\right)\\
=&\sigma\left(\sum_{\lambda'}\int\int x^{(l-1)}\left(2^{-\beta}(u-v+\tilde{u}),\alpha-\beta+\alpha',\lambda'\right)\cdot\right.\\
&\qquad\left.W^{(l)}_{\lambda',\lambda}\left(2^{-\alpha}\tilde{u},\alpha'\right)2^{-2\alpha}d\tilde{u}d\alpha'+b^{(l)}(\lambda)\right)
\end{aligned}$$

and

$$\begin{aligned}
&x^{(l)}[T_{\beta,v}x^{(l-1)}](u,\alpha,\lambda)\\
=&\sigma\left(\sum_{\lambda'}\int\int T_{\beta,v}x^{(l-1)}(u+u',\alpha+\alpha',\lambda')W^{(l)}_{\lambda',\lambda}\left(2^{-\alpha}u',\alpha'\right)2^{-2\alpha}d\alpha'du'+b^{(l)}(\lambda)\right)\\
=&\sigma\left(\sum_{\lambda'}\int\int x^{(l-1)}(2^{-\beta}(u+u'-v),\alpha+\alpha'-\beta,\lambda')\cdot\right.\\
&\qquad\left.W^{(l)}_{\lambda',\lambda}\left(2^{-\alpha}u',\alpha'\right)2^{-2\alpha}d\alpha'du'+b^{(l)}(\lambda)\right)
\end{aligned}$$

Therefore $T_{\beta,v}x^{(l)}[x^{(l-1)}]=x^{(l)}[T_{\beta,v}x^{(l-1)}],\ \forall l>1$.

To prove the necessary part: when $l=1$, we have

$$\begin{aligned}
&T_{\beta,v}x^{(1)}[x^{(0)}](u,\alpha,\lambda)=x^{(1)}[x^{(0)}]\left(2^{-\beta}(u-v),\alpha-\beta,\lambda\right)\\
=&\sigma\left(\sum_{\lambda'}\int x^{(0)}\left(2^{-\beta}(u-v)+u',\lambda'\right)\cdot\right.\\
&\qquad\left.W^{(1)}\left(u',\lambda',2^{-\beta}(u-v),\alpha-\beta,\lambda\right)du'+b^{(1)}(\lambda)\right)\\
=&\sigma\left(\sum_{\lambda'}\int x^{(0)}\left(2^{-\beta}(u+u'-v),\lambda'\right)\cdot\right.
\end{aligned}$$

$$W^{(1)}\left(2^{-\beta}u', \lambda', 2^{-\beta}(u-v), \alpha-\beta, \lambda\right) 2^{-2\beta} du' + b^{(1)}(\lambda)\Big),$$

and

$$\begin{aligned}
&x^{(1)}[D_{\beta,v}x^{(0)}](u,\alpha,\lambda)\\
=&\sigma\left(\sum_{\lambda'}\int D_{\beta,v}x^{(0)}\left(u+u',\lambda'\right) W^{(1)}\left(u',\lambda',u,\alpha,\lambda\right) du' + b^{(1)}(\lambda)\right)\\
=&\sigma\left(\sum_{\lambda'}\int x^{(0)}\left(2^{-\beta}(u+u'-v),\lambda'\right) W^{(1)}\left(u',\lambda',u,\alpha,\lambda\right) du' + b^{(1)}(\lambda)\right)
\end{aligned}$$

Hence for (20) to hold when $l=1$, we need

$$W^{(1)}\left(u',\lambda',u,\alpha,\lambda\right) = W^{(1)}\left(2^{-\beta}u',\lambda',2^{-\beta}(u-v),\alpha-\beta,\lambda\right)2^{-2\beta} \tag{23}$$

to hold true for all $u,\alpha,\lambda,u',\lambda',v,\beta$. Keeping $u,\alpha,\lambda,u',\lambda',\beta$ fixed while changing v in (23), we obtain that $W^{(1)}(u',\lambda',u,\alpha,\lambda)$ does not depend on the third variable u. Thus $W^{(1)}\left(u',\lambda',u,\alpha,\lambda\right) = W^{(1)}\left(u',\lambda',0,\alpha,\lambda\right),\ \forall u$. Define $W^{(1)}_{\lambda,\lambda}(u')$ as

$$W^{(1)}_{\lambda',\lambda}(u') \coloneqq W^{(1)}\left(u',\lambda',0,0,\lambda\right).$$

Then, for any given $u',\lambda',u,\alpha,\lambda$, setting $\beta=\alpha$ in (23) leads to

$$\begin{aligned}
&W^{(1)}\left(u',\lambda',u,\alpha,\lambda\right) = W^{(1)}\left(2^{-\alpha}u',\lambda',2^{-\alpha}(u-v),0,\lambda\right)2^{-2\alpha}\\
=&W^{(1)}\left(2^{-\alpha}u',\lambda',0,0,\lambda\right)2^{-2\alpha} = W^{(1)}_{\lambda',\lambda}\left(2^{-\alpha}u'\right)2^{-2\alpha}.
\end{aligned}$$

Hence (21) can be written as (5).

For $l>1$, a similar argument leads to

$$W^{(l)}\left(u',\alpha',\lambda',u,\alpha,\lambda\right) = W^{(l)}\left(2^{-\beta}u',\alpha',\lambda',2^{-\beta}(u-v),\alpha-\beta,\lambda\right)2^{-2\beta} \tag{24}$$

for all $u,\alpha,\lambda,u',\alpha',\lambda',v,\beta$. Again, keeping $u,\alpha,\lambda,u',\alpha',\lambda',\beta$ fixed while changing v in (24) leads us to the conclusion that $W^{(l)}(u',\alpha',\lambda',u,\alpha,\lambda)$ does not depend on the fourth variable u. Define

$$W^{(l)}_{\lambda',\lambda}(u',\alpha') \coloneqq W^{(l)}\left(u',\alpha',\lambda',0,0,\lambda\right).$$

After setting $\beta=\alpha$ in (24), for any given $u',\alpha',\lambda',u,\alpha,\lambda$, we have

$$\begin{aligned}
&W^{(l)}\left(u',\alpha',\lambda',u,\alpha,\lambda\right) = W^{(l)}\left(2^{-\alpha}u',\alpha',\lambda',2^{-\alpha}(u-v),0,\lambda\right)2^{-2\alpha}\\
=&W^{(l)}\left(2^{-\alpha}u',\alpha',\lambda',0,0,\lambda\right)2^{-2\alpha} = W^{(l)}_{\lambda',\lambda}\left(2^{-\alpha}u'\right)2^{-2\alpha}.
\end{aligned}$$

This concludes the proof of the Theorem. □

A.2 Proof of Proposition 1

Proof of Theorem 1. In a regular $\mathcal{ST}$-equivariant CNN, the l-th convolutional layer (6) is computed as follows:

$$y(u,\alpha,\alpha',\lambda,\lambda') = \int_{\mathbb{R}^2} x^{(l-1)}(u+u',\alpha+\alpha',\lambda') W^{(l)}_{\lambda',\lambda}\left(2^{-\alpha}u',\alpha'\right) 2^{-2\alpha} du', \tag{25}$$

$$z(u,\alpha,\lambda,\lambda') = \int_{\mathbb{R}} y(u,\alpha,\alpha',\lambda,\lambda') d\alpha', \tag{26}$$

$$x^{(l)}(u,\alpha,\lambda) = \sigma\left(\sum_{\lambda'=1}^{M_{l-1}} z(u,\alpha,\lambda,\lambda') + b^{(l)}(\lambda)\right). \tag{27}$$

The spatial convolutions in (25) take $2HWL^2N_sL_\alpha M_l M_{l-1}$ flops (there are $N_sL_\alpha M_l M_{l-1}$ convolutions in u, each taking $2HWL^2$ flops.) The summation over α' in (26) requires $L_\alpha N_s HW M_l M_{l-1}$ flops. The summation over λ', adding the bias, and applying the nonlinear activation in (27) requires an additional $HWN_sM_l(2+M_{l-1})$ flops. Thus the total number of floating point computations in a forward pass through the l-th layer of a regular $\mathcal{ST}$-equivariant CNN is

$$HWN_sM_l(2L^2L_\alpha M_{l-1} + L_\alpha M_{l-1} + M_{l-1} + 2). \tag{28}$$

On the other hand, in an ScDCFNet with separable basis truncation up to KK_α leading coefficients, (6) can be computed via the following steps:

$$y(u,\alpha,\lambda',m) = \int_{\mathbb{R}} x^{(l-1)}(u,\alpha+\alpha',\lambda')\varphi_m(\alpha')d\alpha' \tag{29}$$

$$z(u,\alpha,\lambda',k,m) = \int_{\mathbb{R}^2} y(u+u',\alpha,\lambda',m)\psi_{j_l,k}(2^{-\alpha}u')2^{-2\alpha}du' \tag{30}$$

$$x^{(l)}(u,\alpha,\lambda) = \sigma\left(\sum_{m=1}^{K_\alpha}\sum_{k=1}^{K}\sum_{\lambda'=1}^{M_{l-1}} z(u,\alpha,\lambda',k,m)a^{(l)}_{\lambda',\lambda}(k,m) + b^{(l)}(\lambda)\right). \tag{31}$$

The convolutions in α (29) require $2L_\alpha N_s HW K_\alpha M_{l-1}$ flops (there are $HWK_\alpha M_{l-1}$ convolutions in α, each taking $2L_\alpha N_s$ flops.) The spatial convolutions in (30) take $2HWL^2N_sM_{l-1}K_\alpha K$ flops ($N_sM_{l-1}K_\alpha K$ convolutions in u, each taking $2HWL^2$ flops.) The last step (31) requires an additional $2HWN_sM_l(1+KK_\alpha M_{l-1})$ flops. Hence the total number of floating point computation for an ScDCFNet is

$$2HWN_s(KK_\alpha M_{l-1}M_l + M_l + L^2M_{l-1}K_\alpha K + L_\alpha K_\alpha M_{l-1}). \tag{32}$$

In particular, when $M_l \gg L^2, L_\alpha$, the dominating terms in (28) and (32) are, respectively, $2HWN_sM_lM_{l-1}L^2L_\alpha$ and $2HWN_sM_{l-1}M_lKK_\alpha$. Thus the computational cost in an ScDCFNet has been reduced to a factor of $\frac{KK_\alpha}{L^2L_\alpha}$. □

A.3 Proof of Proposition 2

Before proving Proposition 2, we need the following two lemmas.

Lemma 1. *Suppose that* $\{\psi_k\}_k$ *are the the spatial bases defined in* (8)*, and*

$$F(u) = \sum_k a(k)\psi_{j,k}(u) = \sum_k a(k)2^{-2j}\psi_k(2^{-j}u)$$

is a smooth function on $[-2^j, 2^j]^2$*, then*

$$\int |F(u)|\,du,\ \int |u|\,|\nabla F(u)|\,du,\ 2^j\int |\nabla F(u)|\,du \le \pi\|a\|_\mu = \pi\left(\sum_k \mu_k\cdot a(k)^2\right)^{1/2}. \quad (33)$$

This is essentially Lemma 3.5 and Proposition 3.6 in Qiu et al. (2018) after rescaling u. The only difference is that the basis functions $\psi_k(u)$ defined in (8) are eigenfunctions of Dirichlet Laplacian on $[-1,1]^2$ instead of the unit disk. The key elements of the proof, i.e., the orthogonality of ψ_k and the Poincaré inequality, still apply. Lemma 1 easily leads to the following lemma.

Lemma 2. *Let* $a^{(l)}_{\lambda',\lambda}(k,m)$ *be the coefficients of the filter* $W^{(l)}_{\lambda',\lambda}(u,\alpha)$ *under the joint bases* $\{\psi_k\}_k$ *and* $\{\varphi_m\}_m$ *defined in* (9)*, and define* $W^{(l)}_{\lambda',\lambda,m}(u)$ *as*

$$W_{\lambda',\lambda,m}(u) \coloneqq \sum_k a^{(l)}_{\lambda',\lambda}(k,m)\psi_{j_l,k}(u). \quad (34)$$

We have, for all $l > 1$*,*

$$B^{(1)}_{\lambda',\lambda}, C^{(1)}_{\lambda',\lambda}, 2^{j_1}D^{(1)}_{\lambda',\lambda} \le \pi\|a^{(1)}_{\lambda',\lambda}\|_\mu,\quad B^{(l)}_{\lambda',\lambda,m}, C^{(l)}_{\lambda',\lambda,m}, 2^{j_l}D^{(l)}_{\lambda',\lambda,m} \le \pi\|a^{(l)}_{\lambda',\lambda}(\cdot,m)\|_\mu,$$

where

$$\begin{cases} B^{(1)}_{\lambda',\lambda} \coloneqq \int \left|W^{(1)}_{\lambda',\lambda}(u)\right| du, & B^{(l)}_{\lambda',\lambda,m} \coloneqq \int \left|W^{(l)}_{\lambda',\lambda,m}(u)\right| du,\quad l>1,\\ C^{(1)}_{\lambda',\lambda} \coloneqq \int |u|\left|\nabla_u W^{(1)}_{\lambda',\lambda}(u)\right| du, & C^{(l)}_{\lambda',\lambda,m} \coloneqq \int |u|\left|\nabla_u W^{(l)}_{\lambda',\lambda,m}(u)\right| du,\quad l>1,\\ D^{(1)}_{\lambda',\lambda} \coloneqq \int \left|\nabla_u W^{(1)}_{\lambda',\lambda}(u)\right| du, & D^{(l)}_{\lambda',\lambda,m} \coloneqq \int \left|\nabla_u W^{(l)}_{\lambda',\lambda,m}(u)\right| du,\quad l>1. \end{cases} \quad (35)$$

We thus have

$$B_l, C_l, 2^{j_l}D_l \le A_l,$$

where

$$\begin{aligned} B_1 &\coloneqq \max\left\{\sup_\lambda \sum_{\lambda'=1}^{M_0} B^{(1)}_{\lambda',\lambda},\ \frac{M_0}{M_1}\sup_{\lambda'}\sum_{\lambda=1}^{M_1} B^{(1)}_{\lambda',\lambda}\right\},\\ C_1 &\coloneqq \max\left\{\sup_\lambda \sum_{\lambda'=1}^{M_0} C^{(1)}_{\lambda',\lambda},\ \frac{M_0}{M_1}\sup_{\lambda'}\sum_{\lambda=1}^{M_1} C^{(1)}_{\lambda',\lambda}\right\},\\ D_1 &\coloneqq \max\left\{\sup_\lambda \sum_{\lambda'=1}^{M_0} D^{(1)}_{\lambda',\lambda},\ \frac{M_0}{M_1}\sup_{\lambda'}\sum_{\lambda=1}^{M_1} D^{(1)}_{\lambda',\lambda}\right\}, \end{aligned} \quad (36)$$

and, for $l > 1$,

$$
\begin{aligned}
B_l &\coloneqq \max\left\{\sup_{\lambda}\sum_{\lambda'=1}^{M_{l-1}}\sum_{m} B^{(l)}_{\lambda',\lambda,m},\ \frac{2M_{l-1}}{M_l}\sum_{m} B_{l,m}\right\}, \quad B_{l,m} \coloneqq \sup_{\lambda'}\sum_{\lambda=1}^{M_l} B^{(l)}_{\lambda',\lambda,m},\\
C_l &\coloneqq \max\left\{\sup_{\lambda}\sum_{\lambda'=1}^{M_{l-1}}\sum_{m} C^{(l)}_{\lambda',\lambda,m},\ \frac{2M_{l-1}}{M_l}\sum_{m} C_{l,m}\right\}, \quad C_{l,m} \coloneqq \sup_{\lambda'}\sum_{\lambda=1}^{M_l} C^{(l)}_{\lambda',\lambda,m},\\
D_l &\coloneqq \max\left\{\sup_{\lambda}\sum_{\lambda'=1}^{M_{l-1}}\sum_{m} D^{(l)}_{\lambda',\lambda,m},\ \frac{2M_{l-1}}{M_l}\sum_{m} D_{l,m}\right\}, \quad D_{l,m} \coloneqq \sup_{\lambda'}\sum_{\lambda=1}^{M_l} D^{(l)}_{\lambda',\lambda,m}.
\end{aligned}
\tag{37}
$$

In particular, (A2) implies that $B_l, C_l, 2^{j_l} D_l \le 1,\ \forall l$.

Proof of Proposition 2. To simplify the notation, we omit (l) in $W^{(l)}_{\lambda',\lambda}$, $W^{(l)}_{\lambda',\lambda,m}$, and $b^{(l)}$, and let $M = M_l$, $M' = M_{l-1}$. The proof of (a) for the case $l = 1$ is similar to Proposition 3.1(a) of Qiu et al. (2018) after noticing the fact that

$$
\int_{\mathbb{R}^2} \left|W(2^{-\alpha}u)\right| 2^{-2\alpha} du = \int_{\mathbb{R}^2} |W(u)|\, du, \tag{38}
$$

and we include it here for completeness. From the definition of B_1 in (36), we have

$$
\sup_{\lambda}\sum_{\lambda'} B^{(1)}_{\lambda',\lambda} \le B_1, \quad \text{and} \quad \sup_{\lambda'}\sum_{\lambda} B^{(1)}_{\lambda',\lambda} \le B_1 \frac{M}{M'}.
$$

Thus, given two arbitrary functions x_1 and x_2, we have

$$
\begin{aligned}
&\left|\left(x^{(1)}[x_1] - x^{(1)}[x_2]\right)(u,\alpha,\lambda)\right|^2\\
=&\left|\sigma\left(\sum_{\lambda'}\int x_1(u+u',\lambda') W_{\lambda',\lambda}\left(2^{-\alpha}u'\right) 2^{-2\alpha} du' + b(\lambda)\right)\right.\\
&\left.-\sigma\left(\sum_{\lambda'}\int x_2(u+u',\lambda') W_{\lambda',\lambda}\left(2^{-\alpha}u'\right) 2^{-2\alpha} du' + b(\lambda)\right)\right|^2\\
\le&\left|\sum_{\lambda'}\int x_1(u+u',\lambda') W_{\lambda',\lambda}\left(2^{-\alpha}u'\right) 2^{-2\alpha} du'\right.\\
&\qquad\left.-\sum_{\lambda'}\int x_2(u+u',\lambda') W_{\lambda',\lambda}\left(2^{-\alpha}u'\right) 2^{-2\alpha} du'\right|^2\\
=&\left|\sum_{\lambda'}\int (x_1-x_2)(u+u',\lambda') W_{\lambda',\lambda}\left(2^{-\alpha}u'\right) 2^{-2\alpha} du'\right|^2\\
\le&\left(\sum_{\lambda'}\int \left|(x_1-x_2)(u+u',\lambda')\right|^2 \left|W_{\lambda',\lambda}(2^{-\alpha}u')\right| 2^{-2\alpha} du'\right)\sum_{\lambda'}\int \left|W_{\lambda',\lambda}(2^{-\alpha}u')\right| 2^{-2\alpha} du'
\end{aligned}
$$

$$
\begin{aligned}
&= \left(\sum_{\lambda'} \int \left| (x_1 - x_2)(u + u', \lambda') \right|^2 \left| W_{\lambda',\lambda}(2^{-\alpha} u') \right| 2^{-2\alpha} du' \right) \left(\sum_{\lambda'} B^{(1)}_{\lambda',\lambda} \right) \\
&\leq B_1 \sum_{\lambda'} \int \left| (x_1 - x_2)(\tilde{u}, \lambda') \right|^2 \left| W_{\lambda',\lambda}(2^{-\alpha} (\tilde{u} - u)) \right| 2^{-2\alpha} d\tilde{u}
\end{aligned}
$$

Therefore, for any α,

$$
\begin{aligned}
&\sum_{\lambda} \int \left| \left(x^{(1)}[x_1] - x^{(1)}[x_2] \right) (u, \alpha, \lambda) \right|^2 du \\
\leq &\sum_{\lambda} \int B_1 \sum_{\lambda'} \int \left| (x_1 - x_2)(\tilde{u}, \lambda') \right|^2 \left| W_{\lambda',\lambda}(2^{-\alpha} (\tilde{u} - u)) \right| 2^{-2\alpha} d\tilde{u} du \\
= &B_1 \sum_{\lambda'} \int \left| (x_1 - x_2)(\tilde{u}, \lambda') \right|^2 \left(\sum_{\lambda} \int \left| W_{\lambda',\lambda}(2^{-\alpha} (\tilde{u} - u)) \right| 2^{-2\alpha} du \right) d\tilde{u} \\
= &B_1 \sum_{\lambda'} \int \left| (x_1 - x_2)(\tilde{u}, \lambda') \right|^2 \left(\sum_{\lambda} B^{(1)}_{\lambda',\lambda} \right) d\tilde{u} \\
\leq &B_1^2 \frac{M}{M'} \sum_{\lambda'} \int \left| (x_1 - x_2)(\tilde{u}, \lambda') \right|^2 d\tilde{u} \\
= &B_1^2 M \|x_1 - x_2\|^2 \\
\leq &M \|x_1 - x_2\|^2,
\end{aligned}
$$

where the last inequality makes use of the fact that $B_1 \leq A_1 \leq 1$ under (A2) (Lemma 2.) Therefore

$$
\begin{aligned}
\|x^{(1)}[x_1] - x^{(1)}[x_2]\|^2 &= \sup_{\alpha} \frac{1}{M} \sum_{\lambda} \int \left| \left(x^{(1)}[x_1] - x^{(1)}[x_2] \right) (u, \alpha, \lambda) \right|^2 du \\
&\leq \|x_1 - x_2\|^2.
\end{aligned}
$$

This concludes the proof of (a) for the case $l = 1$. To prove the case for any $l > 1$, we first recall from (37) that

$$
\sup_{\lambda} \sum_{\lambda'} \sum_{m} B^{(l)}_{\lambda',\lambda,m} \leq B_l, \quad \text{and} \quad \sum_{m} B_{l,m} \leq B_l \frac{M}{2M'}, \quad \text{where} \quad B_{l,m} = \sup_{\lambda'} \sum_{\lambda} B^{(l)}_{\lambda',\lambda,m}.
$$

Thus, for two arbitrary functions x_1 and x_2, we have

$$
\begin{aligned}
&\left| \left(x^{(l)}[x_1] - x^{(l)}[x_2] \right) (u, \alpha, \lambda) \right|^2 \\
= &\left| \sigma \left(\sum_{\lambda'} \int_{\mathbb{R}^2} \int_{\mathbb{R}} x_1(u + u', \alpha + \alpha', \lambda') W_{\lambda',\lambda} \left(2^{-\alpha} u', \alpha' \right) 2^{-2\alpha} d\alpha' du' + b(\lambda) \right) \right. \\
&\left. - \sigma \left(\sum_{\lambda'} \int_{\mathbb{R}^2} \int_{\mathbb{R}} x_2(u + u', \alpha + \alpha', \lambda') W_{\lambda',\lambda} \left(2^{-\alpha} u', \alpha' \right) 2^{-2\alpha} d\alpha' du' + b(\lambda) \right) \right|^2
\end{aligned}
$$

$$
\begin{aligned}
&\leq \left| \sum_{\lambda'} \int_{\mathbb{R}^2} \int_{\mathbb{R}} (x_1 - x_2)(u+u', \alpha+\alpha', \lambda') W_{\lambda',\lambda}\left(2^{-\alpha}u', \alpha'\right) 2^{-2\alpha} d\alpha' du' \right|^2 \\
&= \left| \sum_{\lambda'} \int_{\mathbb{R}^2} \int_{\mathbb{R}} (x_1 - x_2)(u+u', \alpha+\alpha', \lambda') 2^{-2\alpha} \sum_m W_{\lambda',\lambda,m}\left(2^{-\alpha}u'\right) \varphi_m(\alpha') d\alpha' du' \right|^2 \\
&= \left| \sum_{\lambda'} \sum_m \int_{\mathbb{R}^2} G_m(u+u', \alpha, \lambda') 2^{-2\alpha} W_{\lambda',\lambda,m}(2^{-\alpha}u') du' \right|^2 \\
&\leq \left(\sum_{\lambda'} \sum_m \int_{\mathbb{R}^2} \left| G_m(u+u', \alpha, \lambda') \right|^2 \left| W_{\lambda',\lambda,m}(2^{-\alpha}u') \right| 2^{-2\alpha} du' \right) \cdot \\
&\qquad \left(\sum_{\lambda'} \sum_m \int_{\mathbb{R}^2} \left| W_{\lambda',\lambda,m}(2^{-\alpha}u') \right| 2^{-2\alpha} du' \right) \\
&= \left(\sum_{\lambda'} \sum_m \int_{\mathbb{R}^2} \left| G_m(\tilde{u}, \alpha, \lambda') \right|^2 \left| W_{\lambda',\lambda,m}(2^{-\alpha}(\tilde{u}-u)) \right| 2^{-2\alpha} d\tilde{u} \right) \left(\sum_{\lambda'} \sum_m B^{(l)}_{\lambda',\lambda,m} \right) \\
&\leq B_l \sum_{\lambda'} \sum_m \int_{\mathbb{R}^2} \left| G_m(\tilde{u}, \alpha, \lambda') \right|^2 \left| W_{\lambda',\lambda,m}(2^{-\alpha}(\tilde{u}-u)) \right| 2^{-2\alpha} d\tilde{u},
\end{aligned}
$$

where

$$
G_m(u, \alpha, \lambda') \coloneqq \int_{\mathbb{R}} (x_1 - x_2)(u, \alpha+\alpha', \lambda') \varphi_m(\alpha') d\alpha'. \tag{39}
$$

We claim (to be proved later in Lemma 3) that

$$
M' \|G_m\|^2 = \sup_{\alpha} \sum_{\lambda'} \int_{\mathbb{R}^2} \left| G_m(u, \alpha, \lambda') \right|^2 du \leq 2M' \|x_1 - x_2\|^2, \quad \forall m.
$$

Thus, for any α,

$$
\begin{aligned}
&\sum_{\lambda} \int_{\mathbb{R}^2} \left| \left(x^{(l)}[x_1] - x^{(l)}[x_2] \right)(u, \alpha, \lambda) \right|^2 du \\
\leq &\sum_{\lambda} \int_{\mathbb{R}^2} B_l \sum_{\lambda'} \sum_m \int_{\mathbb{R}^2} \left| G_m(\tilde{u}, \alpha, \lambda') \right|^2 \left| W_{\lambda',\lambda,m}(2^{-\alpha}(\tilde{u}-u)) \right| 2^{-2\alpha} d\tilde{u} du \\
= &B_l \sum_{\lambda'} \sum_m \int_{\mathbb{R}^2} \left| G_m(\tilde{u}, \alpha, \lambda') \right|^2 \left(\sum_{\lambda} \int_{\mathbb{R}^2} \left| W_{\lambda',\lambda,m}(2^{-\alpha}(\tilde{u}-u)) \right| 2^{-2\alpha} du \right) d\tilde{u} \\
= &B_l \sum_{\lambda'} \sum_m \int_{\mathbb{R}^2} \left| G_m(\tilde{u}, \alpha, \lambda') \right|^2 \left(\sum_{\lambda} B^{(l)}_{\lambda',\lambda,m} \right) d\tilde{u} \\
\leq &B_l \sum_{\lambda'} \sum_m \int_{\mathbb{R}^2} \left| G_m(\tilde{u}, \alpha, \lambda') \right|^2 B_{l,m} d\tilde{u} \\
= &B_l \sum_m \left(\sum_{\lambda'} \int_{\mathbb{R}^2} \left| G_m(\tilde{u}, \alpha, \lambda') \right|^2 d\tilde{u} \right) B_{l,m}
\end{aligned}
$$

$$\leq B_l \cdot 2M' \|x_1 - x_2\|^2 \sum_m B_{l,m}$$
$$\leq B_l^2 \cdot 2M' \|x_1 - x_2\|^2 \frac{M}{2M'} \leq M\|x_1 - x_2\|^2.$$

Therefore

$$\|x^{(l)}[x_1] - x^{(l)}[x_2]\|^2 = \sup_\alpha \frac{1}{M} \sum_\lambda \int_{\mathbb{R}^2} \left| \left(x^{(l)}[x_1] - x^{(l)}[x_2] \right)(u, \alpha, \lambda) \right|^2 du$$
$$\leq \|x_1 - x_2\|^2.$$

To prove (b), we use the method of induction. When $l = 0$, $x_0^{(0)}(u, \lambda) = 0$ by definition. When $l = 1$, $x_0^{(1)}(u, \alpha, \lambda) = \sigma(b^{(1)}(\lambda))$. Suppose $x_0^{(l-1)}(u, \alpha, \lambda) = x_0^{(l-1)}(\lambda)$ for some $l > 1$, then

$$\begin{aligned}
&x_0^{(l)}(u, \alpha, \lambda) \\
=&\sigma\left(\sum_{\lambda'} \int_{\mathbb{R}^2} \int_{\mathbb{R}} x_0^{(l-1)}(u + u', \alpha + \alpha', \lambda') W_{\lambda',\lambda}^{(l)} \left(2^{-\alpha} u', \alpha'\right) 2^{-2\alpha} d\alpha' du' + b^{(l)}(\lambda) \right) \\
=&\sigma\left(\sum_{\lambda'} x_0^{(l-1)}(\lambda') \int_{\mathbb{R}^2} \int_{\mathbb{R}} W_{\lambda',\lambda}^{(l)} \left(2^{-\alpha} u', \alpha'\right) 2^{-2\alpha} d\alpha' du' + b^{(l)}(\lambda) \right) \\
=&\sigma\left(\sum_{\lambda'} x_0^{(l-1)}(\lambda') \int_{\mathbb{R}^2} \int_{\mathbb{R}} W_{\lambda',\lambda}^{(l)} \left(u', \alpha'\right) d\alpha' du' + b^{(l)}(\lambda) \right) \\
=&x_0^{(l)}(\lambda).
\end{aligned}$$

Part (c) is an easy corollary of part (a). More specifically, for any $l > 1$,

$$\|x_c^{(l)}\| = \|x^{(l)} - x_0^{(l)}\| = \|x^{(l)}[x^{(l-1)}] - x_0^{(l)}[x_0^{(l-1)}]\| \leq \|x^{(l-1)} - x_0^{(l-1)}\| = \|x_c^{(l-1)}\|.$$

□

Lemma 3. *Suppose $\varphi \in L^2(\mathbb{R})$ with $\operatorname{supp}(\varphi_m) \subset [-1, 1]$ and $\|\varphi\|_{L^2} = 1$, and x is a function of three variables*

$$x : \mathbb{R}^2 \times \mathbb{R} \times [M] \to \mathbb{R}$$
$$(u, \alpha, \lambda) \mapsto x(u, \alpha, \lambda)$$

with $\|x\|^2 \coloneqq \sup_\alpha \frac{1}{M} \sum_\lambda \int_{\mathbb{R}^2} |x(u, \alpha, \lambda)|^2 du$. Define $G(u, \alpha, \lambda)$ as

$$G(u, \alpha, \lambda) \coloneqq \int_{\mathbb{R}} x(u, \alpha + \alpha', \lambda) \varphi(\alpha') d\alpha.$$

Then we have

$$M\|G\|^2 = \sup_\alpha \sum_\lambda \int_{\mathbb{R}^2} |G(u, \alpha, \lambda)|^2 du \leq 2M\|x\|^2. \tag{40}$$

Proof of Lemma 3. Notice that, for any α, we have

$$\begin{aligned}\sum_{\lambda}\int_{\mathbb{R}^2}|G(u,\alpha,\lambda)|^2\,du &= \sum_{\lambda}\int_{\mathbb{R}^2}\left|\int_{-1}^{1}x(u,\alpha+\alpha',\lambda)\varphi(\alpha')d\alpha'\right|^2 du\\ &\le \sum_{\lambda}\int_{\mathbb{R}^2}\left(\int_{-1}^{1}\left|x(u,\alpha+\alpha',\lambda)\right|^2 d\alpha'\right)\|\varphi\|_{L^2}^2 du\\ &= \int_{-1}^{1}\left(\sum_{\lambda}\int_{\mathbb{R}^2}\left|x(u,\alpha+\alpha',\lambda)\right|^2 du\right)d\alpha'\\ &\le \int_{-1}^{1}M\|x\|^2 d\alpha' = 2M\|x\|^2.\end{aligned}$$

Thus

$$\sup_{\alpha}\sum_{\lambda}\int_{\mathbb{R}^2}|G(u,\alpha,\lambda)|^2\,du \le 2M\|x\|^2.$$

□

A.4 Proof of Theorem 2

To prove Theorem 2, we need the following two Propositions.

Proposition 3. *In an ScDCFNet satisfying (A1) and (A3), we have*

(a) For any $l \ge 1$,

$$\left\|x^{(l)}[D_\tau x^{(l-1)}] - D_\tau x^{(l)}[x^{(l-1)}]\right\| \le 4(B_l + C_l)|\nabla\tau|_\infty\|x_c^{(l-1)}\|. \tag{41}$$

(b) For any $l \ge 1$, we have

$$\|T_{\beta,v}x^{(l)}\| = 2^\beta\|x^{(l)}\|, \tag{42}$$

and

$$\left\|x^{(l)}[T_{\beta,v}\circ D_\tau x^{(l-1)}] - T_{\beta,v}D_\tau x^{(l)}[x^{(l-1)}]\right\| \le 2^{\beta+2}(B_l + C_l)|\nabla\tau|_\infty\|x_c^{(l-1)}\|, \tag{43}$$

where the first $T_{\beta,v}$ in (43) is replaced by $D_{\beta,v}$ when $l = 1$.

(c) If (A2) also holds true, then

$$\left\|x^{(l)}[D_{\beta,v}\circ D_\tau x^{(0)}] - T_{\beta,v}D_\tau x^{(l)}[x^{(0)}]\right\| \le 2^{\beta+3}l|\nabla\tau|_\infty\|x^{(0)}\|, \quad \forall l \ge 1. \tag{44}$$

Proposition 4. *In an ScDCFNet satisfying (A1) and (A3), we have, for any $l \ge 1$,*

$$\left\|T_{\beta,v}D_\tau x^{(l)} - T_{\beta,v}x^{(l)}\right\| \le 2^{\beta+1}|\tau|_\infty D_l\|x_c^{(l-1)}\| \le 2^{\beta+1}|\tau|_\infty D_l\|x^{(0)}\|. \tag{45}$$

If (A2) also holds true, then

$$\left\|T_{\beta,v}D_\tau x^{(l)} - T_{\beta,v}x^{(l)}\right\| \le 2^{\beta+1-j_l}|\tau|_\infty\|x^{(0)}\|. \tag{46}$$

Proof of Theorem 2. Putting together (44) and (46), we have

$$\begin{aligned}
&\left\|x^{(L)}[D_{\beta,v}\circ D_\tau x^{(0)}]-T_{\beta,v}x^{(L)}[x^{(0)}]\right\|\\
\leq&\left\|x^{(L)}[D_{\beta,v}\circ D_\tau x^{(0)}]-T_{\beta,v}D_\tau x^{(L)}[x^{(0)}]\right\|+\left\|T_{\beta,v}D_\tau x^{(L)}[x^{(0)}]-T_{\beta,v}x^{(L)}[x^{(0)}]\right\|\\
\leq&2^{\beta+3}L|\nabla\tau|_\infty\|x^{(0)}\|+2^{\beta+1-j_L}|\tau|_\infty\|x^{(0)}\|\\
=&2^{\beta+1}\left(4L|\nabla\tau|_\infty+2^{-j_L}|\tau|_\infty\right)\|x^{(0)}\|
\end{aligned}$$

This concludes the proof of Theorem 2. □

Finally, we need to prove Proposition 3 and Proposition 4, where the following lemma from Qiu et al. (2018) is useful.

Lemma 4 (Lemma A.1 of Qiu et al. (2018))**.** *Suppose that* $|\nabla\tau|_\infty<1/5, \rho(u)=u-\tau(u)$*, then at every point* $u\in\mathbb{R}^2$*,*

$$||J\rho|-1|\leq|\nabla\tau|_\infty(2+|\nabla\tau|_\infty), \tag{47}$$

where $J\rho$ *is the Jacobian of* ρ*, and* $|J\rho|$ *is the Jacobian determinant. As a result,*

$$||J\rho|-1|,\left||J\rho^{-1}|-1\right|\leq 4|\nabla\tau|_\infty, \tag{48}$$

and,

$$|J\rho|,\left|J\rho^{-1}\right|\leq 2. \tag{49}$$

Proof of Proposition 3. Just like Proposition 2(a), the proof of Proposition 3(a) for the case $l=1$ is similar to Lemma 3.2 of Qiu et al. (2018) after the change of variable (38). We thus focus only on the proof for the case $l>1$. To simplify the notation, we denote $x^{(l)}[x^{(l-1)}]$ as $y[x]$, and replace $x_c^{(l-1)}$, $W^{(l)}$, $b^{(l)}$, M_{l-1}, and M_l, respectively, by x_c, W, b, M', and M. By the definition of the deformation D_τ (12), we have

$$\begin{aligned}
D_\tau y[x](u,\alpha,\lambda)&=\sigma\left(\sum_{\lambda'}\int_{\mathbb{R}^2}\int_{\mathbb{R}}x(\rho(u)+u',\alpha+\alpha',\lambda')W_{\lambda',\lambda}\left(2^{-\alpha}u',\alpha'\right)2^{-2\alpha}d\alpha'du'+b(\lambda)\right),\\
y[D_\tau x](u,\alpha,\lambda)&=\sigma\left(\sum_{\lambda'}\int_{\mathbb{R}^2}\int_{\mathbb{R}}x(\rho(u+u'),\alpha+\alpha',\lambda')W_{\lambda',\lambda}\left(2^{-\alpha}u',\alpha'\right)2^{-2\alpha}d\alpha'du'+b(\lambda)\right).
\end{aligned}$$

Thus

$$\begin{aligned}
&|(D_\tau y[x]-y[D_\tau x])(u,\alpha,\lambda)|^2\\
=&\left|\sigma\left(\sum_{\lambda'}\int_{\mathbb{R}^2}\int_{\mathbb{R}}x(\rho(u)+u',\alpha+\alpha',\lambda')W_{\lambda',\lambda}\left(2^{-\alpha}u',\alpha'\right)2^{-2\alpha}d\alpha'du'+b(\lambda)\right)\right.\\
&\left.-\sigma\left(\sum_{\lambda'}\int_{\mathbb{R}^2}\int_{\mathbb{R}}x(\rho(u+u'),\alpha+\alpha',\lambda')W_{\lambda',\lambda}\left(2^{-\alpha}u',\alpha'\right)2^{-2\alpha}d\alpha'du'+b(\lambda)\right)\right|^2
\end{aligned}$$

$$
\begin{aligned}
\leq & \left| \sum_{\lambda'} \int_{\mathbb{R}^2} \int_{\mathbb{R}} \left(x(\rho(u)+u', \alpha+\alpha', \lambda') - x(\rho(u)+u', \alpha+\alpha', \lambda') \right) \cdot \right. \\
& \qquad \left. W_{\lambda',\lambda}\left(2^{-\alpha}u', \alpha'\right) 2^{-2\alpha} d\alpha' du' \right|^2 \\
= & \left| \sum_{\lambda'} \int_{\mathbb{R}^2} \int_{\mathbb{R}} \left(x_c(\rho(u)+u', \alpha+\alpha', \lambda') - x_c(\rho(u)+u', \alpha+\alpha', \lambda') \right) \cdot \right. \\
& \qquad \left. W_{\lambda',\lambda}\left(2^{-\alpha}u', \alpha'\right) 2^{-2\alpha} d\alpha' du' \right|^2 \\
= & \left| \sum_{\lambda'} \sum_{m} \int_{\mathbb{R}^2} \int_{\mathbb{R}} \left(x_c(\rho(u)+u', \alpha+\alpha', \lambda') - x_c(\rho(u)+u', \alpha+\alpha', \lambda') \right) \cdot \right. \\
& \qquad \left. W_{\lambda',\lambda,m}\left(2^{-\alpha}u'\right) \varphi_m\left(\alpha'\right) 2^{-2\alpha} d\alpha' du' \right|^2,
\end{aligned}
$$

where the second equality results from the fact that $x(u,\alpha,\lambda) - x_c(u,\alpha,\lambda) = x_0(\lambda)$ depends only on λ (Proposition 2(b).) Just like the proof of Proposition 2(a), we take the integral of α' first, and define

$$
G_m(u,\alpha,\lambda') \coloneqq \int_{\mathbb{R}} x_c(u, \alpha+\alpha', \lambda') \varphi_m(\alpha') d\alpha'. \tag{50}
$$

Thus

$$
\begin{aligned}
& |(D_\tau y[x] - y[D_\tau x])(u,\alpha,\lambda)|^2 \\
\leq & \left| \sum_{\lambda'} \sum_{m} \int_{\mathbb{R}^2} \left(G_m(\rho(u)+u', \alpha, \lambda') - G_m(\rho(u+u'), \alpha, \lambda') \right) \cdot \right. \\
& \qquad \left. W_{\lambda',\lambda,m}\left(2^{-\alpha}u'\right) 2^{-2\alpha} du' \right|^2 \\
= & \left| \sum_{\lambda'} \sum_{m} \int_{\mathbb{R}^2} G_m(v,\alpha,\lambda') W_{\lambda',\lambda,m}\left(2^{-\alpha}(v-\rho(u))\right) 2^{-2\alpha} dv \right. \\
& \left. - \sum_{\lambda'} \sum_{m} \int_{\mathbb{R}^2} G_m(v,\alpha,\lambda') W_{\lambda',\lambda,m}\left(2^{-\alpha}(\rho^{-1}(v)-u)\right) 2^{-2\alpha} |J\rho^{-1}(v)| dv \right|^2 \\
= & \left| E_1(u,\alpha,\lambda) + E_2(u,\alpha,\lambda) \right|^2,
\end{aligned}
$$

where

$$
\begin{aligned}
E_1(u,\alpha,\lambda) = & \sum_{\lambda'} \sum_{m} \int_{\mathbb{R}^2} \left[W_{\lambda',\lambda,m}\left(2^{-\alpha}(v-\rho(u))\right) - W_{\lambda',\lambda,m}\left(2^{-\alpha}(\rho^{-1}(v)-u)\right) \right] \cdot \\
& \qquad 2^{-2\alpha} G_m(v,\alpha,\lambda') dv, \\
E_2(u,\alpha,\lambda) = & \sum_{\lambda'} \sum_{m} \int_{\mathbb{R}^2} W_{\lambda',\lambda,m}\left(2^{-\alpha}(\rho^{-1}(v)-u)\right) \left[1 - \left| J\rho^{-1}(v) \right| \right] \cdot \\
& \qquad 2^{-2\alpha} G_m(v,\alpha,\lambda') dv.
\end{aligned}
$$

Therefore

$$
M \left\| D_\tau y[x] - y[D_\tau x] \right\|^2 = \sup_{\alpha} \sum_{\lambda} \int_{\mathbb{R}^2} |(D_\tau y[x] - y[D_\tau x])(u,\alpha,\lambda)|^2 \, du
$$

$$\begin{aligned}&\leq \sup_{\alpha} \sum_{\lambda} \int_{\mathbb{R}^2} |E_1(u,\alpha,\lambda) + E_2(u,\alpha,\lambda)|^2 \, du \\ &= M \|E_1 + E_2\|^2\end{aligned}$$

Hence

$$\|D_\tau y[x] - y[D_\tau x]\| \leq \|E_1 + E_2\|. \tag{51}$$

We thus seek to estimate $\|E_1\|$ and $\|E_2\|$ individually.

To bound $\|E_2\|$, we let

$$k^{(2)}_{\lambda',\lambda,m}(v,u,\alpha) \coloneqq W_{\lambda',\lambda,m}\left(2^{-\alpha}(\rho^{-1}(v)-u)\right)\left[1-|J\rho^{-1}(v)|\right]2^{-2\alpha}.$$

Then

$$E_2(u,\alpha,\lambda) = \sum_{\lambda'}\sum_{m}\int_{\mathbb{R}^2} G_m(v,\alpha,\lambda') k^{(2)}_{\lambda',\lambda,m}(v,u,\alpha)dv,$$

and, for any given v and α

$$\begin{aligned}\int_{\mathbb{R}^2}\left|k^{(2)}_{\lambda',\lambda,m}(v,u,\alpha)\right|du &= \int_{\mathbb{R}^2}\left|W_{\lambda',\lambda,m}\left(2^{-\alpha}(\rho^{-1}(v)-u)\right)\right|\left|1-|J\rho^{-1}(v)|\right|2^{-2\alpha}du \\ &= \left|1-|J\rho^{-1}(v)|\right|\int_{\mathbb{R}^2}\left|W_{\lambda',\lambda,m}(\tilde{u})\right|d\tilde{u} \\ &\leq 4|\nabla\tau|_\infty B^{(l)}_{\lambda',\lambda,m},\end{aligned}$$

where the last inequality comes from (48). Moreover, for any given u and α,

$$\begin{aligned}\int_{\mathbb{R}^2}\left|k^{(2)}_{\lambda',\lambda,m}(v,u,\alpha)\right|dv &= \int_{\mathbb{R}^2}\left|W_{\lambda',\lambda,m}\left(2^{-\alpha}(\rho^{-1}(v)-u)\right)\right|\left|1-|J\rho^{-1}(v)|\right|2^{-2\alpha}dv \\ &= \int_{\mathbb{R}^2}\left|W_{\lambda',\lambda,m}(\tilde{v}-2^{-\alpha}u)\right|\cdot\left||J\rho(2^\alpha\tilde{v})|-1\right|d\tilde{v} \\ &\leq 4|\nabla\tau|_\infty B^{(l)}_{\lambda',\lambda,m},\end{aligned}$$

where the last inequality is again because of (48). Thus, for any given α,

$$\begin{aligned}&\sum_{\lambda}\int_{\mathbb{R}^2}|E_2(u,\alpha,\lambda)|^2\,du = \sum_{\lambda}\int_{\mathbb{R}^2}\left|\sum_{\lambda'}\sum_{m}\int_{\mathbb{R}^2}G_m(v,\alpha,\lambda')k^{(2)}_{\lambda',\lambda,m}(v,u,\alpha)dv\right|^2 du \\ \leq &\sum_{\lambda}\int_{\mathbb{R}^2}\left(\sum_{\lambda'}\sum_{m}\int_{\mathbb{R}^2}|G_m(v,\alpha,\lambda)|^2\left|k^{(2)}_{\lambda',\lambda,m}(v,u,\alpha)\right|dv\right)\cdot \\ &\qquad\left(\sum_{\lambda'}\sum_{m}\int_{\mathbb{R}^2}\left|k^{(2)}_{\lambda',\lambda,m}(v,u,\alpha)\right|dv\right)du \\ \leq &\sum_{\lambda}\int_{\mathbb{R}^2}\left(\sum_{\lambda'}\sum_{m}\int_{\mathbb{R}^2}|G_m(v,\alpha,\lambda)|^2\left|k^{(2)}_{\lambda',\lambda,m}(v,u,\alpha)\right|dv\right)\cdot\end{aligned}$$

$$
\begin{aligned}
&\left(\sum_{\lambda'}\sum_{m} 4|\nabla\tau|_\infty B^{(l)}_{\lambda',\lambda,m}\right) du \\
\leq & 4|\nabla\tau|_\infty B_l \sum_m \sum_{\lambda'} \int_{\mathbb{R}^2} |G_m(v,\alpha,\lambda)|^2 \left(\sum_\lambda \int_{\mathbb{R}^2} \left|k^{(2)}_{\lambda',\lambda,m}(v,u,\alpha)\right| du\right) dv \\
\leq & 4|\nabla\tau|_\infty B_l \sum_m \sum_{\lambda'} \int_{\mathbb{R}^2} |G_m(v,\alpha,\lambda)|^2 \left(\sum_\lambda 4|\nabla\tau|_\infty B^{(l)}_{\lambda',\lambda,m}\right) dv \\
\leq & 16|\nabla\tau|^2_\infty B_l \sum_m \left(\sum_{\lambda'} \int_{\mathbb{R}^2} |G_m(v,\alpha,\lambda)|^2 dv\right) B_{l,m} \\
\leq & 16|\nabla\tau|^2_\infty B_l \sum_m M'\|G_m\|^2 B_{l,m}
\end{aligned}
$$

Since $\|G_m\|^2 \leq 2\|x_c\|^2$ (by Lemma 3), and $\sum_m B_{l,m} \leq \frac{M}{2M'} B_l$ by definition (37), we thus have, for any α,

$$
\sum_\lambda \int_{\mathbb{R}^2} |E_2(u,\alpha,\lambda)|^2 du \leq 16|\nabla\tau|^2_\infty B_l \frac{M}{2M'} B_l \cdot 2M'\|x_c\|^2 = M(4|\nabla\tau|_\infty B_l \|x_c\|)^2. \tag{52}
$$

Taking $\sup_\alpha$ on both sides gives us

$$
\|E_2\| \leq 4|\nabla\tau|_\infty B_l \|x_c\|. \tag{53}
$$

Similarly, to bound $\|E_1\|$, we introduce

$$
k^{(1)}_{\lambda',\lambda,m}(v,u,\alpha) \coloneqq \left[W_{\lambda',\lambda,m}\left(2^{-\alpha}(v-\rho(u))\right) - W_{\lambda',\lambda,m}\left(2^{-\alpha}(\rho^{-1}(v)-u)\right)\right] 2^{-2\alpha}.
$$

Then

$$
E_1(u,\alpha,\lambda) = \sum_{\lambda'}\sum_m \int_{\mathbb{R}^2} G_m(v,\alpha,\lambda') k^{(1)}_{\lambda',\lambda,m}(v,u,\alpha) dv,
$$

and, for any given v and α, we have

$$
\int_{\mathbb{R}^2} \left|k^{(1)}_{\lambda',\lambda,m}(v,u,\alpha)\right| du, \quad \int_{\mathbb{R}^2} \left|k^{(1)}_{\lambda',\lambda,m}(v,u,\alpha)\right| dv \leq 4|\nabla\tau|_\infty C^{(l)}_{\lambda',\lambda,m}. \tag{54}
$$

The proof of (54) is exactly the same as that of Lemma 3.2 in Qiu et al. (2018) after a change of variable, and we thus omit the detail. Similar to the procedure we take to bound $\|E_2\|$, (54) leads to

$$
\|E_1\| \leq 4|\nabla\tau|_\infty C_l \|x_c\|. \tag{55}
$$

Putting together (51), (53), and (55), we thus have

$$
\|D_\tau y[x] - y[D_\tau x]\| \leq \|E_1 + E_2\| \leq \|E_1\| + \|E_2\| \leq 4(B_l + C_l)|\nabla\tau|_\infty \|x_c\|.
$$

This concludes the proof of (a).

To prove (b), given any $\beta \in \mathbb{R}$, and $v \in \mathbb{R}^2$, we have

$$\begin{aligned}
\|T_{\beta,v}x^{(l)}\|^2 &= \sup_{\alpha} \frac{1}{M_l} \sum_{\lambda} \int_{\mathbb{R}^2} \left| T_{\beta,v}x^{(l)}(u,\alpha,\lambda) \right|^2 du \\
&= \sup_{\alpha} \frac{1}{M_l} \sum_{\lambda} \int_{\mathbb{R}^2} \left| x^{(l)}(2^{-\beta}(u-v),\alpha-\beta,\lambda) \right|^2 du \\
&= \sup_{\alpha} \frac{1}{M_l} \sum_{\lambda} \int_{\mathbb{R}^2} \left| x^{(l)}(\tilde{u},\alpha-\beta,\lambda) \right|^2 2^{2\beta} d\tilde{u} \\
&= 2^{2\beta}\|x^{(l)}\|^2
\end{aligned}$$

Thus (42) holds true. As for (43), we have

$$\begin{aligned}
&\left\| x^{(l)}[T_{\beta,v} \circ D_\tau x^{(l-1)}] - T_{\beta,v} D_\tau x^{(l)}[x^{(l-1)}] \right\| \\
=&\left\| T_{\beta,v} x^{(l)}[D_\tau x^{(l-1)}] - T_{\beta,v} D_\tau x^{(l)}[x^{(l-1)}] \right\| \\
=&2^\beta \left\| x^{(l)}[D_\tau x^{(l-1)}] - D_\tau x^{(l)}[x^{(l-1)}] \right\| \\
\leq&2^{\beta+2}(B_l + C_l)|\nabla\tau|_\infty \|x_c^{(l-1)}\|,
\end{aligned}$$

where the first equality holds valid because of Theorem 1, and the second equality comes from (42).

To prove (c), for any $0 \leq j \leq l$, define y_j as

$$y_j = x^{(l)} \circ x^{(l-1)} \circ \cdots \circ T_{\beta,v} \circ D_\tau x^{(j)} \circ \cdots \circ x^{(0)}.$$

We thus have

$$\begin{aligned}
&\left\| x^{(l)}[D_{\beta,v} \circ D_\tau x^{(0)}] - T_{\beta,v} D_\tau x^{(l)}[x^{(0)}] \right\| = \|y_l - y_0\| \leq \sum_{j=1}^{l} \|y_j - y_{j-1}\| \\
=&\sum_{j=1}^{l} \left\| x^{(l)} \circ \cdots \circ T_{\beta,v} \circ D_\tau x^{(j)} \circ \cdots \circ x^{(0)} - x^{(l)} \circ \cdots \circ x^{(j)} \circ T_{\beta,v} \circ D_\tau x^{(j-1)} \circ \cdots \circ x^{(0)} \right\| \\
\leq&\sum_{j=1}^{l} \left\| T_{\beta,v} \circ D_\tau x^{(j)}[x^{(j-1)}] - x^{(j)}[T_{\beta,v} \circ D_\tau x^{(j-1)}] \right\| \\
\leq&\sum_{j=1}^{l} 2^{\beta+2}(B_j + C_j)|\nabla\tau|_\infty \|x_c^{(j-1)}\| \\
\leq&\sum_{k=1}^{l} 2^{\beta+2} \cdot 2|\nabla\tau|_\infty \|x^{(0)}\| = 2^{\beta+3} l |\nabla\tau|_\infty \|x^{(0)}\|,
\end{aligned}$$

where the second inequality is because of Proposition 2(a), the third inequality is due to (43), and the last inequality holds true because $B_l, C_l \leq A_l \leq 1$ under (A2) (Lemma 2.) This concludes the proof of Proposition 3. □

Proof of Proposition 4. The second inequality in (45) is due to Proposition 2(c). Because of (42), the first inequality in (45) is equivalent to

$$\left\| D_\tau x^{(l)} - x^{(l)} \right\| \leq 2|\tau|_\infty D_l \|x_c^{(l-1)}\| \tag{56}$$

Just like Proposition 3(a), the proof of (56) for the case $l = 1$ is similar to Proposition 3.4 of Qiu et al. (2018) after the change of variable (38). A similar strategy as that of Proposition 3(a) can be used to extend the proof to the case $l > 1$. More specifically, denote $x^{(l-1)}, x_c^{(l-1)}, W^{(l)}, b^{(l)}$, respectively, as x, x_c, W, and b to simplify the notation. We have

$$\begin{aligned}
&\left| \left(D_\tau x^{(l)}[x] - x^{(l)}[x] \right) (u, \alpha, \lambda) \right|^2 \\
=& \left| \sigma \left(\sum_{\lambda'} \int_{\mathbb{R}^2} \int_{\mathbb{R}} x \left(\rho(u) + u', \alpha + \alpha', \lambda' \right) W_{\lambda',\lambda} \left(2^{-\alpha} u', \alpha' \right) 2^{-2\alpha} du' d\alpha' + b(\lambda) \right) \right. \\
& \left. - \sigma \left(\sum_{\lambda'} \int_{\mathbb{R}^2} \int_{\mathbb{R}} x \left(u + u', \alpha + \alpha', \lambda' \right) W_{\lambda',\lambda} \left(2^{-\alpha} u', \alpha' \right) 2^{-2\alpha} du' d\alpha' + b(\lambda) \right) \right| \\
\leq& \left| \sum_{\lambda'} \int_{\mathbb{R}^2} \int_{\mathbb{R}} x \left(\rho(u) + u', \alpha + \alpha', \lambda' \right) W_{\lambda',\lambda} \left(2^{-\alpha} u', \alpha' \right) 2^{-2\alpha} du' d\alpha' \right. \\
& \left. - \sum_{\lambda'} \int_{\mathbb{R}^2} \int_{\mathbb{R}} x \left(u + u', \alpha + \alpha', \lambda' \right) W_{\lambda',\lambda} \left(2^{-\alpha} u', \alpha' \right) 2^{-2\alpha} du' d\alpha' \right| \\
=& \left| \sum_{\lambda'} \int_{\mathbb{R}^2} \int_{\mathbb{R}} x_c \left(\rho(u) + u', \alpha + \alpha', \lambda' \right) \sum_m W_{\lambda',\lambda,m} \left(2^{-\alpha} u' \right) \varphi_m \left(\alpha' \right) 2^{-2\alpha} du' d\alpha' \right. \\
& \left. - \sum_{\lambda'} \int_{\mathbb{R}^2} \int_{\mathbb{R}} x_c \left(u + u', \alpha + \alpha', \lambda' \right) \sum_m W_{\lambda',\lambda,m} \left(2^{-\alpha} u' \right) \varphi_m \left(\alpha' \right) 2^{-2\alpha} du' d\alpha' \right| \\
=& \left| \sum_{\lambda'} \sum_m \int_{\mathbb{R}^2} G_m(v, \alpha, \lambda') k_{\lambda',\lambda,m}(v, u, \alpha) du' \right|,
\end{aligned}$$

where

$$\begin{aligned}
G_m(u, \alpha, \lambda') &\coloneqq \int_{\mathbb{R}} x_c(u, \alpha + \alpha', \lambda') \varphi_m(\alpha') d\alpha', \\
k_{\lambda',\lambda,m}(v, u, \alpha) &\coloneqq 2^{-2\alpha} \left[W_{\lambda',\lambda,m} \left(2^{-\alpha}(v - \rho(u)) \right) - W_{\lambda',\lambda,m} \left(2^{-\alpha}(v - u) \right) \right].
\end{aligned}$$

Similar to (54), we have the following bound

$$\int_{\mathbb{R}^2} \left| k_{\lambda',\lambda,m}(v, u, \alpha) \right| du, \quad \int_{\mathbb{R}^2} \left| k_{\lambda',\lambda,m}(v, u, \alpha) \right| dv \leq 2|\nabla \tau|_\infty D_{\lambda',\lambda,m}^{(l)}. \tag{57}$$

Again, the proof of (57) is the same as that of Proposition 3.4 in Qiu et al. (2018) after a change of variable. The rest of the proof follows from a similar argument as in (52) and (53). □

A.5 Proof of Theorem 3

Before proving Theorem 3, we need the following lemma.

Lemma 5. *Under the same assumption of Theorem 3, we have, for any l,*

$$\left\|\tilde{x}^{(l)}[x^{(0)}] - x^{(l)}[x^{(0)}]\right\|^2 = \sup_{\alpha \le T} \frac{1}{M_l} \sum_{\lambda=1}^{M_l} \int \left|\tilde{x}^{(l)}(u,\alpha,\lambda) - x^{(l)}(u,\alpha,\lambda)\right|^2 du = O(2^{-T}), \quad (58)$$

where we slightly abuse the notation $\tilde{x}^{(l)}$ and $x^{(l)}$ to denote the l-th layer outputs given the input $x^{(0)}$ in the first equality.

Proof of Lemma 5. We prove this lemma by induction. When $l = 1$, we have, for any $\alpha \in [-T, T]$,

$$\tilde{x}^{(1)}(u,\alpha,\lambda) = x^{(1)}(u,\alpha,\lambda). \quad (59)$$

When $\alpha \le -T$, we have

$$\begin{aligned}
&\left|\tilde{x}^{(1)}(u,\alpha,\lambda) - x^{(1)}(u,\alpha,\lambda)\right| = \left|x^{(1)}(u,-T,\lambda) - x^{(1)}(u,\alpha,\lambda)\right| \\
=&\left|\sigma\left(\sum_{\lambda'} \int_{\mathbb{R}^2} 2^{2T} x^{(0)}(u+u',\lambda') W_{\lambda',\lambda}(2^T u') du' + b(\lambda)\right)\right. \\
&\left. -\sigma\left(\sum_{\lambda'} \int_{\mathbb{R}^2} 2^{-2\alpha} x^{(0)}(u+u',\lambda') W_{\lambda',\lambda}(2^{-2\alpha} u') du' + b(\lambda)\right)\right| \\
\le&\left|\sum_{\lambda'} x^{(0)}(\cdot,\lambda') * W_{\lambda',\lambda,-T}(u) - x^{(0)}(\cdot,\lambda') * W_{\lambda',\lambda,\alpha}(u)\right|,
\end{aligned}$$

where $W_{\lambda',\lambda,\alpha}(u) = 2^{-2\alpha} W_{\lambda',\lambda}(2^{-\alpha}u)$ forms a mollifier in $\mathbb{R}^2$. Thus, we have

$$\begin{aligned}
&\int_{\mathbb{R}^2} \left|\tilde{x}^{(1)}(u,\alpha,\lambda) - x^{(1)}(u,\alpha,\lambda)\right|^2 du \\
\le&\int_{\mathbb{R}^2} \left|\sum_{\lambda'} x^{(0)}(\cdot,\lambda') * W_{\lambda',\lambda,-T}(u) - x^{(0)}(\cdot,\lambda') * W_{\lambda',\lambda,\alpha}(u)\right|^2 du \\
\le&\int_{\mathbb{R}^2} \left(\sum_{\lambda'} \left|x^{(0)}(\cdot,\lambda') * W_{\lambda',\lambda,-T}(u) - x^{(0)}(\cdot,\lambda') * W_{\lambda',\lambda,\alpha}(u)\right|^2\right) M' du \\
=&M' \sum_{\lambda'} \left\|x^{(0)}(\cdot,\lambda') * W_{\lambda',\lambda,-T} - x^{(0)}(\cdot,\lambda') * W_{\lambda',\lambda,\alpha}\right\|_2^2 \\
\le&M' \sum_{\lambda'} \left(\left\|x^{(0)}(\cdot,\lambda') * W_{\lambda',\lambda,-T} - A_{\lambda',\lambda} x^{(0)}(\cdot,\lambda')\right\|_2\right. \\
&\qquad \left. + \left\|x^{(0)}(\cdot,\lambda') * W_{\lambda',\lambda,\alpha} - A_{\lambda',\lambda} x^{(0)}(\cdot,\lambda')\right\|_2\right)^2,
\end{aligned}$$

where $A_{\lambda',\lambda} = \int_{\mathbb{R}^2} W^{(1)}_{\lambda',\lambda}(u)du$. Due to the L^2 convergence of mollification (Evans, 2010), i.e.,

$$\left\| f * W_{\lambda',\lambda,\alpha} - A_{\lambda',\lambda} f \right\|_2 \leq C 2^{\alpha} \left\| f \right\|_{H^1} = O(2^{\alpha}), \quad \text{as } \alpha \to -\infty,$$

we have

$$\sup_{\alpha \leq -T} \int_{\mathbb{R}^2} \left| \tilde{x}^{(1)}(u,\alpha,\lambda) - x^{(1)}(u,\alpha,\lambda) \right|^2 du = O(2^{-T}). \tag{60}$$

Combining (60) and (59), we have

$$\|\tilde{x}^{(1)} - x^{(1)}\|^2 = \sup_{\alpha \leq T} \frac{1}{M} \sum_{\lambda} \int \left| \tilde{x}^{(1)}(u,\alpha,\lambda) - x^{(1)}(u,\alpha,\lambda) \right|^2 du = O(2^{-T}).$$

Next, we want to show that (58) holds for $l > 1$ assuming it holds for $l-1$. Indeed, for any $\alpha \in [-T, T]$, we have

$$\tilde{x}^{(l)}(u,\alpha,\lambda) - x^{(l)}(u,\alpha,\lambda) = x^{(l)}[\tilde{x}^{(l-1)}](u,\alpha,\lambda) - x^{(l)}[x^{(l-1)}](u,\alpha,\lambda).$$

Taking the supremum over $\alpha \in [-T, T]$, we have

$$\begin{aligned}
&\sup_{\alpha \in [-T,T]} \frac{1}{M} \sum_{\lambda} \int \left| \tilde{x}^{(l)}(u,\alpha,\lambda) - x^{(l)}(u,\alpha,\lambda) \right|^2 du \\
=&\sup_{\alpha \in [-T,T]} \frac{1}{M} \sum_{\lambda} \int \left| x^{(l)}[\tilde{x}^{(l-1)}](u,\alpha,\lambda) - x^{(l)}[x^{(l-1)}](u,\alpha,\lambda) \right|^2 du \\
\leq& \left\| x^{(l)}[\tilde{x}^{(l-1)}] - x^{(l)}[x^{(l-1)}] \right\|^2 \\
\leq& \left\| \tilde{x}^{(l-1)} - x^{(l-1)} \right\|^2 = O(2^{-T}),
\end{aligned} \tag{61}$$

where the last inequality results from the non-expansiveness of $x^{(l)}$ (c.f. Proposition 2,) and the last equality comes from our assumption that (58) holds for $l-1$. For any $\alpha \leq -T$, we have

$$\begin{aligned}
&\frac{1}{M} \sum_{\lambda} \int \left| \tilde{x}^{(l)}(u,\alpha,\lambda) - x^{(l)}(u,\alpha,\lambda) \right|^2 du \\
=&\frac{1}{M} \sum_{\lambda} \int \left| \tilde{x}^{(l)}(u,-T,\lambda) - x^{(l)}(u,\alpha,\lambda) \right|^2 du \\
=&\frac{1}{M} \sum_{\lambda} \int \left| \tilde{x}^{(l)}(u,-T,\lambda) - x^{(l)}(u,-T,\lambda) + x^{(l)}(u,-T,\lambda) - x^{(l)}(u,\alpha,\lambda) \right|^2 du \\
\leq&\frac{1}{M} \sum_{\lambda} \int \left| \tilde{x}^{(l)}(u,-T,\lambda) - x^{(l)}(u,-T,\lambda) + x^{(l)}(u,-T,\lambda) - x^{(l)}(u,\alpha,\lambda) \right|^2 du \\
\leq&\frac{2}{M} \sum_{\lambda} \int \left| \tilde{x}^{(l)}(u,-T,\lambda) - x^{(l)}(u,-T,\lambda) \right|^2 + \left| x^{(l)}(u,-T,\lambda) - x^{(l)}(u,\alpha,\lambda) \right|^2 du \\
=&O(2^{-T}) + \frac{2}{M} \sum_{\lambda} \int \left| x^{(l)}(u,-T,\lambda) - x^{(l)}(u,\alpha,\lambda) \right|^2 du,
\end{aligned}$$

where, in the last equality, the first term $O(2^{-T})$ does not depend on $\alpha \leq -T$ and comes from (61). Taking the supremum over $\alpha \leq -T$, we have

$$\begin{aligned}
&\sup_{\alpha\leq -T} \frac{1}{M}\sum_{\lambda}\int \left|\tilde{x}^{(l)}(u,\alpha,\lambda) - x^{(l)}(u,\alpha,\lambda)\right|^2 du \\
\leq& O(2^{-T}) + \sup_{\alpha\leq -T} \frac{2}{M}\sum_{\lambda}\int \left|x^{(l)}(u,-T,\lambda) - x^{(l)}(u,\alpha,\lambda)\right|^2 du \\
=& O(2^{-T}),
\end{aligned} \tag{62}$$

where the last equality holds for the same reason as (60). Combining (61) and (62), we thus have

$$\left\|\tilde{x}^{(l)}[x^{(0)}] - x^{(l)}[x^{(0)}]\right\|^2 = \sup_{\alpha\leq T}\frac{1}{M_l}\sum_{\lambda=1}^{M_l}\int \left|\tilde{x}^{(l)}(u,\alpha,\lambda) - x^{(l)}(u,\alpha,\lambda)\right|^2 du = O(2^{-T}).$$

□

With Lemma 5 proven, the proof of Theorem 3 is merely an easy application of triangle inequality:

Proof of Theorem 3.

$$\begin{aligned}
&\left\|\tilde{x}^{(L)}[D_{\beta,v}\circ D_\tau x^{(0)}] - T_{\beta,v}\tilde{x}^{(L)}[x^{(0)}]\right\| \\
\leq& \left\|\tilde{x}^{(L)}[D_{\beta,v}\circ D_\tau x^{(0)}] - x^{(L)}[D_{\beta,v}\circ D_\tau x^{(0)}]\right\| + \left\|x^{(L)}[D_{\beta,v}\circ D_\tau x^{(0)}] - T_{\beta,v}x^{(L)}[x^{(0)}]\right\| \\
&+ \left\|T_{\beta,v}x^{(L)}[x^{(0)}] - T_{\beta,v}\tilde{x}^{(L)}[x^{(0)}]\right\| \\
\leq& O(2^{-T}) + 2^{\beta+1}\left(4L|\nabla\tau|_\infty + 2^{-j_L}|\tau|_\infty\right)\|x^{(0)}\| + 2^\beta O(2^{-T}) \\
=& 2^{\beta+1}\left(4L|\nabla\tau|_\infty + 2^{-j_L}|\tau|_\infty\right)\|x^{(0)}\| + O(2^{-T}),
\end{aligned} \tag{63}$$

where the last inequality (63) comes from Lemma 5 and Theorem 2. □

Appendix B. Experimental Details in Section 6

We explain in this appendix the experimental details in Section 6 of the main text.

B.1 Verification of $\mathcal{ST}$-equivariance

The ScDCFNet used in this experiment has two convolutional layers, each of which is composed of a $\mathcal{ST}$-equivariant convolution (5) or (6), a batch-normalization, and a 2×2 spatial average-pooling. The expansion coefficients $a^{(1)}_{\lambda',\lambda}(k)$ and $a^{(2)}_{\lambda',\lambda}(k,m)$ are sampled from a Gaussian distribution and truncated to $K=8$ and $K_\alpha = 3$ leading coefficients for u and α respectively. Similarly, a regular CNN with two convolutional layers and randomly generated 5×5 convolutional kernels is used as a baseline for comparison.

B.2 Image Reconstruction

The network architectures for the ScDCFNet and regular CNN auto-encoders are shown in Table 5. The filter expansion in the ScDCFNet auto-encoder is truncated to $K = 8$ and $K_\alpha = 3$. SGD with decreasing learning rate from 10^{-2} to 10^{-4} is used to train both networks for 20 epochs.

Layer	Regular auto-encoder	ScDCF auto-encoder
1	c7x7x1x8 ReLU ap2x2	sc(15)13x13x1x4 ReLU ap2x2
2	c7x7x8x16 ReLU ap2x2	sc(15)13x13x3x4x8 ReLU ap2x2
3	fc128 ReLU fc4096 ReLU	fc128 ReLU fc4096 ReLU
4	ct7x7x16x8 ReLU us2x2	ct7x7x16x8 ReLU us2x2
5	ct7x7x8x1 ReLU us2x2	ct7x7x8x1 ReLU us2x2

Table 5: Architectures of the auto-encoders used for the experiment in Section 6.4. The encoded representation is the output of the second layer. **cLxLxM'xM:** a regular convolutional layer with M' input channels, M output channels, and LxL spatial kernels. **apLxL:** LxL average-pooling. **sc(N_s)LxLxM'xM:** the first-layer convolution (5) in ScDCFNet, where N_s is the number of scale channels, and LxL is the spatial kernel size. **sc(N_s)LxLxL_αxM'xM:** the l-th layer ($l > 1$) convolution (6) in ScDCFNet, where the extra symbol L_α stands for the filter size in α. **fcM:** a fully connected layer with M output channels. **ctLxLxM'xM:** transposed-convolutional layers with M' input channels, M output channels, and LxL spatial kernels. **us2x2**: 2x2 spatial upsampling. Batch-normalization (not shown in the table) is used after each convolutional layer.

References

Milton Abramowitz and Irene A Stegun. *Handbook of mathematical functions: with formulas, graphs, and mathematical tables*, volume 55. Courier Corporation, 1965.

Vincent Andrearczyk, Julien Fageot, Valentin Oreiller, Xavier Montet, and Adrien Depeursinge. Exploring local rotation invariance in 3D CNNs with steerable filters. In *International Conference on Medical Imaging with Deep Learning*, pages 15–26. PMLR, 2019.

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

Erik Johannes Bekkers, Marco Loog, Bart M ter Haar Romeny, and Remco Duits. Template matching via densities on the roto-translation group. *IEEE transactions on pattern analysis and machine intelligence*, 40(2):452–466, 2017.

Alberto Bietti and Julien Mairal. Invariance and stability of deep convolutional representations. In *NIPS 2017-31st Conference on Advances in Neural Information Processing Systems*, pages 1622–1632, 2017.

Alberto Bietti and Julien Mairal. Group invariance, stability to deformations, and complexity of deep convolutional representations. *The Journal of Machine Learning Research*, 20(1): 876–924, 2019.

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

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.

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*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011.

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

Taco Cohen and Max Welling. Steerable CNNs. In *International Conference on Learning Representations*, 2017.

Taco S. Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Spherical CNNs. In *International Conference on Learning Representations*, 2018.

Taco S Cohen, Mario Geiger, and Maurice Weiler. A general theory of equivariant cnns on homogeneous spaces. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Carlos Esteves, Christine Allen-Blanchette, Ameesh Makadia, and Kostas Daniilidis. Learning SO(3) equivariant representations with spherical CNNs. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–68, 2018.

Lawrence C. Evans. *Partial differential equations*. American Mathematical Society, Providence, R.I., 2010. ISBN 9780821849743 0821849743.

William T. Freeman and Edward H Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (9):891–906, 1991.

Rohan Ghosh and Anupam K Gupta. Scale steerable filters for locally scale-invariant convolutional neural networks. *arXiv preprint arXiv:1906.03861*, 2019.

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*, pages 770–778, 2016.

Emiel Hoogeboom, Jorn W.T. Peters, Taco S. Cohen, and Max Welling. Hexaconv. In *International Conference on Learning Representations*, 2018.

Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Weinberger. Multi-scale dense networks for resource efficient image classification. In *International Conference on Learning Representations*, 2018.

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.

Dunham Jackson. *The theory of approximation*, volume 11. American Mathematical Soc., 1930.

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

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

Tsung-Wei Ke, Michael Maire, and Stella X Yu. Multigrid neural architectures. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6665–6673, 2017.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Risi Kondor and Shubhendu Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2747–2755. PMLR, 10–15 Jul 2018.

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

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

Xiang Li, Wenhai Wang, Xiaolin Hu, and Jian Yang. Selective kernel networks. 2019.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

Julien Mairal. End-to-end kernel learning with supervised convolutional kernel networks. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

Julien Mairal, Piotr Koniusz, Zaid Harchaoui, and Cordelia Schmid. Convolutional kernel networks. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

Stéphane Mallat. Recursive interferometric representation. In *Proc. of EUSICO conference, Danemark*, 2010.

Stéphane Mallat. Group invariant scattering. *Communications on Pure and Applied Mathematics*, 65(10):1331–1398, 2012.

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.

Diego Marcos, Benjamin Kellenberger, Sylvain Lobry, and Devis Tuia. Scale equivariance in CNNs with vector fields. *arXiv preprint arXiv:1807.11783*, 2018.

Daniël M Pelt and James A Sethian. A mixed-scale dense convolutional neural network for image analysis. *Proceedings of the National Academy of Sciences*, 115(2):254–259, 2018.

Qiang Qiu, Xiuyuan Cheng, Robert Calderbank, and Guillermo Sapiro. DCFNet: Deep neural network with decomposed convolutional filters. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4198–4207, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing. ISBN 978-3-319-24574-4.

Laurent Sifre and Stéphane Mallat. Rotation, scaling and deformation invariant scattering for texture discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1233–1240, 2013.

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

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv preprint arXiv:1802.08219*, 2018.

Panqu Wang, Pengfei Chen, Ye Yuan, Ding Liu, Zehua Huang, Xiaodi Hou, and Garrison Cottrell. Understanding convolution for semantic segmentation. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1451–1460. IEEE, 2018.

Maurice Weiler and Gabriele Cesa. General e(2)-equivariant steerable cnns. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Maurice Weiler, Mario Geiger, Max Welling, Wouter Boomsma, and Taco Cohen. 3d steerable cnns: Learning rotationally equivariant features in volumetric data. In *Advances in Neural Information Processing Systems*, pages 10381–10392, 2018a.

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*, pages 849–858, 2018b.

Marysia Winkels and Taco S Cohen. 3D G-CNNs for pulmonary nodule detection. *arXiv preprint arXiv:1804.04656*, 2018.

Daniel Worrall and Gabriel Brostow. Cubenet: Equivariance to 3D rotation and translation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 567–584, 2018.

Daniel Worrall and Max Welling. Deep scale-spaces: Equivariance over scale. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 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*, pages 5028–5037, 2017.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

Yichong Xu, Tianjun Xiao, Jiaxing Zhang, Kuiyuan Yang, and Zheng Zhang. Scale-invariant convolutional neural networks. *arXiv preprint arXiv:1411.6369*, 2014.

Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *International Conference on Learning Representations*, 2016.

Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. Dilated residual networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

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