PDO-eConvs: Partial Differential Operator Based Equivariant Convolutions

Zhengyang Shen [1] Lingshen He [2] Zhouchen Lin [2] Jinwen Ma [1]

Abstract

Recent research has shown that incorporating equivariance into neural network architectures is very helpful, and there have been some works investigating the equivariance of networks under group actions. However, as digital images and feature maps are on the discrete meshgrid, corresponding equivariance-preserving transformation groups are very limited.

In this work, we deal with this issue from the connection between convolutions and partial differential operators (PDOs). In theory, assuming inputs to be smooth, we transform PDOs and propose a system which is equivariant to a much more general continuous group, the n-dimension Euclidean group. In implementation, we discretize the system using the numerical schemes of PDOs, deriving approximately equivariant convolutions (PDO-eConvs). Theoretically, the approximation error of PDO-eConvs is of the quadratic order. It is the first time that the error analysis is provided when the equivariance is approximate. Extensive experiments on rotated MNIST and natural image classification show that PDO-eConvs perform competitively yet use parameters much more efficiently. Particularly, compared with Wide ResNets, our methods result in better results using only 12.6% parameters.

1. Introduction

In the past few years, convolutional neural network (CNN) models have become the dominant machine learning methods in the field of computer vision for various tasks, such as image recognition, objective detection and semantic segmentation. Compared with fully-connected neural networks, a significant advantage of CNNs is that they are shift equivariant: shifting an image and then feeding it through a number of layers is the same as feeding the original image and then shifting the resulted feature maps. In other words, the translation symmetry is preserved by each layer. Also, the equivariance property brings in weight sharing, with which we can use parameters more efficiently.

Motivated by this, Cohen and Welling (2016) proposed group equivariant CNNs (G-CNNs), showing how convolutional networks can be generalized to exploit larger groups of symmetries, including rotations and reflections. G-CNNs are equivariant to the group $p4m$ or $p4$[1], and work on square lattices. In addition, Hoogeboom et al. (2018) proposed HexaConv and showed how one can implement planar convolutions and group convolutions over hexagonal lattices, instead of square ones. As a result, the equivariance is expanded to $p6m$. However, it seems impossible to design CNNs that are equivariant to the rotation angles other than $\pi/2$ ($p4m$) and $\pi/3$ ($p6m$) as there does not seem to exist other rotational symmetric discrete lattices on the 2D plane, if one considers equivariance in the ways as (Cohen & Welling, 2016) and (Hoogeboom et al., 2018).

In order to exploit more symmetries, Weiler et al. (2018) employed harmonics as steerable filters to achieve exact equivariance to larger transformation groups in the continuous domain. However, they are difficult to preserve strong equivariance when operating on discrete pixel grids, for two main reasons: (i) When a harmonic is sampled on grids with a low rate, it could appear as a lower harmonic, which introduces aliasing artifacts. (ii) With Gaussian radial profiles as radial functions, harmonics ranged out of the sampled kernel support, leading to a high equivariance error on implementation.

From another point of view, a conventional convolutional filter can also be viewed as a linear combination of PDOs, which was proposed by (Ruthotto & Haber, 2018). With this new understanding, we assume inputs are smooth functions, and then show how to transform the PDOs and get a system which is exactly equivariant to a much more general continu-

[1]School of Mathematical Sciences and LMAM, Peking University, Beijing 100871 [2]Key Lab. of Machine Perception (MoE), School of EECS, Peking University, Beijing 100871. Correspondence to: Zhouchen Lin <zlin@pku.edu.cn>, Jinwen Ma <jwma@math.pku.edu.cn>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 108, 2020. Copyright 2020 by the author(s).

[1]Generally, the group pnm, which we will use in Section 4, denotes the group generated by translations, reflections and rotations by $2\pi/n$. The group pn denotes the group only generated by translations and rotations by $2\pi/n$.

ous transformation group, the n-dimension Euclidean group. To implement our theory on discrete digital images, we discretize the system using the numerical schemes of PDOs and get approximately equivariant convolutions. Particularly, the discretized convolutions can achieve a quadratic order equivariance approximation, and it is the first time that the error analysis is provided when the equivariance is approximate. As the derived equivariant convolutions are based on PDOs, we refer to them as PDO-eConvs.

We evaluate the performance of PDO-eConvs on rotated MNIST and natural image classification tasks. Extensive experiments verify that PDO-eConv produces very competitive results and is significantly efficient on parameter learning..

Our contributions are as follows:

- With the assumption that inputs are smooth, we use PDOs to design a system that is equivariant to a much more general continuous group, the n-dimensional Euclidean group.
- The equivariance is exact in the continuous domain. It becomes approximate only after the discretization. Moreover, it is the first time that the error analysis is provided when the equivariance is approximate. To be specific, the approximation error of PDO-eConvs is of the quadratic order, indicating a precise approximation.
- Extensive experiments on PDO-eConvs show that our methods perform competitively and have significant parameter efficiency.

2. Prior and Related Work

2.1. Equivariant CNNs

Lenc & Vedaldi (2015) showed that the AlexNet CNN (Krizhevsky et al., 2012) trained on ImageNet spontaneously learned representations that are equivariant to flips, scalings and rotations, which supported the idea that equivariance is a good inductive bias for CNNs. Cohen & Welling (2016; 2017) succeeded in incorporating equivariance into neural networks. However, these methods can only deal with a 4-fold rotational symmetry for images with square pixels. Hoogeboom et al. (2018) alleviated this limit by implementing planar convolutions and group convolutions over hexagonal lattices. Consequently, they can deal with a 6-fold rotational symmetry.

Since there does not seem to have more rotational symmetries on lattices in the 2D plane, some works designed approximately equivariant networks w.r.t. larger groups. Zhou et al. (2017) and Marcos et al. (2017) utilized bilinear interpolation to help produce feature maps at different orientations. They are inherently approximately equivariant. By comparison, ours is exactly equivariant in the continuous domain. Worral et al. (2017) used harmonics to extract features and achieve equivariance to 360-rotation, but the equivariance is destroyed after Gaussion-resampling. Weiler et al. (2018) and Cesa & Weiler (2019) employed harmonics as steerable filters to achieve exact equivariance w.r.t. larger groups in the continuous domain, but the equivariance is difficult to preserve in the discrete domain due to aliasing artifacts and limited kernel support. So they used much larger filters to achieve approximate equivariance, resulting in CNNs with a large computational burden. By contrast, PDO-eConvs can use a relatively small kernel size to achieve theoretically guaranteed exact equivariance in the discrete domain, which makes big difference.

There are also some empirical approaches for enforcing equivariance. A commonly utilized technique is data augmentation, see e.g. (Krizhevsky et al., 2012). The basic idea is to enrich the training set by transformed samples. Laptev et al. (2016) used parallel siamese architectures for the considered transformation set and applying the transformation-invariant pooling (TI-Pooling) operator on their outputs. Jaderberg et al. (2015) applied a differentiable module to actively transform feature maps, and then Esteves et al. (2018) used this method to help enforce equivariance under rotation and scale transformations. In (Sabour et al., 2017; Hinton et al., 2018), capsules are used to represent the location information and enforce equivariance. However, these methods learn the transformations directly from datas, which are inferior to those methods incorporating equivariance into architectures for lack of interpretability and reliability.

2.2. The Relationship between Convolutions and PDOs

There have been extensive works (Jain & Jain, 1978; Witkin, 1987; Koenderink, 1984; Perona & Malik, 1990; Osher & Rudin, 1990) utilizing PDOs to process images. The relationship between convolutions and PDOs was presented in (Dong et al., 2017; Ruthotto & Haber, 2018), where the authors translated convolutional filters to linear combinations of PDOs, and this approximation has good analytical properties. Some works (Long et al., 2018; 2019) used this new understanding to help design CNNs. Also, this relationship is an important theoretical foundation of our work.

Actually, there exist some works using PDOs to investigate equivariance. Liu et al. (2013) designed a partial differential equation (PDE) using a linear combination of equivariant PDOs and proposed learning based PDEs, which are naturally shift and rotation equivariant. Fang et al. (2017) further adopted this technique on face recognition task. However, the capacity of learning based PDEs cannot be compared with that of nowadays widely used CNNs.

3. Mathematical Framework

In this section we design a group equivariant system using PDOs. To make concepts and notations more explicit, we give a preliminary introduction of groups and equivariance formally.

3.1. Prior Knowledge

The Isometry Group In mathematics, the isometry group is a group consisted of isometry transformations, which preserve the distance of any two points. Particularly, the Euclidean group is the largest isometry group defined on $\mathbb{R}^n$, which we denote as $E(n)$. Given $y \in \mathbb{R}^n$, the isometry transformation is:

$$y :\rightarrow Ay + x, \tag{1}$$

where A is an orthogonal matrix, i.e., $A^\top A = I$, and $x \in \mathbb{R}^n$. When $A = I$, the transformations in (1) compose the translation group $\langle \mathbb{R}^n; + \rangle$. Without ambiguity, we use $\mathbb{R}^n$ to denote the translation group in the following text. When $x = 0$, $E(n)$ degenerates to the orthogonal group, $O(n)$, which contains all the orthogonal transformations, including reflections and rotations. We use A to parameterize $O(n)$. $\mathbb{R}^n$ and $O(n)$ are both subgroups of $E(n)$, and $E(n) = \mathbb{R}^n \rtimes O(n)$ ($\rtimes$ is a semidirect-product). We use (x, A) to represent the element in $E(n)$, where x and A represent a translation and an orthogonal transformation, respectively. Restricting the domain of A and x, we can also use this representation to parametrize any subgroup of $E(n)$.

Actions on Functions Inputs and intermediate feature maps can be naturally modeled as functions defined in the continuous domain. To be specific, we model the input r as a smooth function defined on $\mathbb{R}^n$ and the intermediate feature map e as a smooth function defined on $E(n)$, where the smoothness of e means that if we use the representation (x, A) mentioned above, the feature map $e(x, A)$ is smooth w.r.t. x when A is fixed. So e can also be viewed as a function defined on $\mathbb{R}^n$ with infinite channels indexed by A. We use $C^\infty(\mathbb{R}^n)$ and $C^\infty(E(n))$[2] to denote the function spaces of r and e, respectively .

In this way, transformations like rotations and reflections on inputs and feature maps can be mathematically formulated. Here, we introduce two transformations used in our theory.

- Suppose that $r \in C^\infty(\mathbb{R}^n)$ and $\widetilde{A} \in O(n)$, then the transfomation $\widetilde{A}$ acts on r in the following way[3]:

$$\forall x \in \mathbb{R}^n, \quad \pi^R_{\widetilde{A}}[r](x) = r(\widetilde{A}^{-1}x). \tag{2}$$

[2]For the simplicity of our theory, we require that $r \in C^\infty(\mathbb{R}^n)$. However, in implementation, we only require that $r \in C^4(\mathbb{R}^n)$. The requirement on e is the same.

[3]We use $[\cdot]$ to denote that an operator acts on a function.

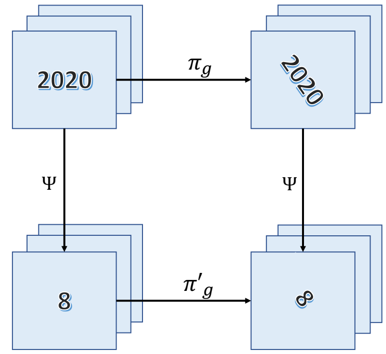

Figure 1. The transformation g can be preserved by the mapping Ψ.

- Suppose that $e \in C^\infty(E(n))$ and $\widetilde{A} \in O(n)$, then $\widetilde{A}$ acts on e in the following way:

$$\forall a \in E(n), \quad \pi^E_{\widetilde{A}}[e](a) = e(\widetilde{A}^{-1}a), \tag{3}$$

where $\widetilde{A}^{-1}a$ is group product on $E(n)$. Using the representation of $E(n)$, it is of the following more detailed form:

$$\pi^E_{\widetilde{A}}[e](x, A) = e(\widetilde{A}^{-1}x, \widetilde{A}^{-1}A), \tag{4}$$

where (x, A) is the representation of a.

Equivariance Equivariance measures how the outputs of a mapping transform in a predictable way with the transformation of the inputs. Here, we formulate it in detail. Let Ψ be a mapping from the input feature space to the output feature space and G is a group. A group equivariant Ψ satisfies that

$$\forall g \in G, \quad \Psi[\pi_g[f]] = \pi'_g[\Psi[f]],$$

where f can be any input feature map in the input feature space, and π_g and π'_g denote how the transformation g acts on input features and output features, respectively.

That is, transforming an input f by a transformation g (forming $\pi_g[f]$) and then passing it through the mapping Ψ should give the same result as first mapping f through Ψ and then transforming the representation. The schema of equivariance is shown in Figure 1. It is easy to see that if each layer of a network is equivariant, the equivariance can be preserved by the network.

3.2. Group Equivariant Differential Operators

We refer to $H(u_1, u_2, \cdots, u_n; \boldsymbol{\beta})$ as a polynomial of n variables parameterized by $\boldsymbol{\beta}$. $\frac{\partial}{\partial x_i}$ denotes the derivative with respect to the ith coordinate of x. Obviously, as a polynomial of PDOs $\left\{\frac{\partial}{\partial x_i}\right\}_{i=1}^n$, $H(\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}, \ldots, \frac{\partial}{\partial x_n}; \boldsymbol{\beta})$ is a linear combination of PDOs parameterized by $\boldsymbol{\beta}$. For example,

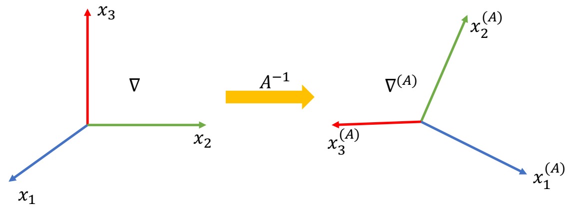

Figure 2. Transformation over coordinate frame.

if $H(u_1, u_2; \boldsymbol{\beta}) = \beta_1 u_1 + \beta_2 u_1 u_2$, then $H(\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}; \boldsymbol{\beta}) = \beta_1 \frac{\partial}{\partial x_1} + \beta_2 \frac{\partial^2}{\partial x_1 \partial x_2}$.

3.2.1. Under Orthogonal Transformation

We transform these PDOs with orthogonal matrices, and define the following differential operator:

$$\chi^{(A)} = H\left(\frac{\partial}{\partial x_1^{(A)}}, \frac{\partial}{\partial x_2^{(A)}} \cdots, \frac{\partial}{\partial x_n^{(A)}}; \boldsymbol{\beta}\right), \tag{5}$$

where

$$\begin{bmatrix} \frac{\partial}{\partial x_1^{(A)}} \\ \frac{\partial}{\partial x_2^{(A)}} \\ \vdots \\ \frac{\partial}{\partial x_n^{(A)}} \end{bmatrix} = A^{-1} \begin{bmatrix} \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} \\ \vdots \\ \frac{\partial}{\partial x_n} \end{bmatrix}, \tag{6}$$

and A is an orthogonal matrix. As a compact format, we can also rewrite (6) as

$$\nabla^{(A)} = A^{-1} \nabla, \tag{7}$$

where $\nabla = [\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}, \cdots, \frac{\partial}{\partial x_n}]^T$, which is a gradient operator. Particularly, the canonical operator $\chi^{(I)} = H(\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}, \cdots, \frac{\partial}{\partial x_n}; \boldsymbol{\beta})$. From another point of view, the transformation on PDOs can also be viewed as that we transform the coordinate frame according to A, and then conduct differential operators on the new coordinate frame (see Figure 2). Particularly, PDOs can be viewed as steerable filters in the sense of (Helor & Teo, 1996), because the transformed versions of PDOs can be expressed as linear combinations of PDOs.

Next, we employ $\chi^{(A)}$'s to define two differential operators Ψ and Φ. To be specific, we use Ψ to deal with inputs, which maps an input $r \in C^\infty(\mathbb{R}^n)$ to a feature map defined on $E(n)$: $\forall (x, A) \in E(n)$,

$$\forall (x, A) \in E(n), \quad \Psi[r](x, A) = \chi^{(A)}[r](x). \tag{8}$$

Then we use Φ to deal with the resulting feature maps, which maps one feature map $e \in C^\infty(E(n))$ to another feature map defined on $E(n)$: $\forall (x, A) \in E(n)$,

$$\Phi[e](x, A) = \int_{O(n)} \chi_B^{(A)}\ [e](x, AB) d\nu(B), \tag{9}$$

where B is an orthogonal matrix and ν is a measure on $O(n)$. As for $\chi_B^{(A)}$, we use the subscript B to distinguish the differential operators parameterized by different $\boldsymbol{\beta}_B$'s. The e on the right hand side should be viewed as a function defined on $\mathbb{R}^n$ indexed by AB when the operator $\chi_B^{(A)}$ acts on it.

We now show that the above two operators are equivariant under orthogonal transformations and describe how the outputs transform w.r.t. the transformations of inputs.

Theorem 1 *If $r \in C^\infty(\mathbb{R}^n), e \in C^\infty(E(n))$ and $\widetilde{A} \in O(n)$, the following rules are satisfied:*

$$\Psi\left[\pi^R_{\widetilde{A}}[r]\right] = \pi^E_{\widetilde{A}}\left[\Psi[r]\right], \tag{10}$$

$$\Phi\left[\pi^E_{\widetilde{A}}[e]\right] = \pi^E_{\widetilde{A}}\left[\Phi[e]\right], \tag{11}$$

where $\pi^R_{\widetilde{A}}, \pi^E_{\widetilde{A}}, \Psi$ and Φ are defined in (2), (4), (8) and (9), respectively.

Proof 1 *To prove (10), we need to prove that $\forall x \in \mathbb{R}^n, A \in O(n)$,*

$$\begin{aligned} \chi^{(A)}\left[\pi^R_{\widetilde{A}}[r]\right](x) &= \pi^E_{\widetilde{A}}\left[\chi^{(A)}[r](x)\right] \\ &= \chi^{(\widetilde{A}^{-1}A)}[r](\widetilde{A}^{-1}x). \end{aligned} \tag{12}$$

We first show that

$$\begin{aligned} \nabla^{(A)}\left[\pi^R_{\widetilde{A}}[r]\right](x) &= (A^{-1}\nabla)\left[\pi^R_{\widetilde{A}}[r]\right](x) \\ &= (A^{-1}\nabla)\left[r(\widetilde{A}^{-1}x)\right] \\ &= A^{-1}\widetilde{A}\nabla[r](\widetilde{A}^{-1}x) \\ &= (\widetilde{A}^{-1}A)^{-1}\nabla[r](\widetilde{A}^{-1}x) \\ &= \nabla^{(\widetilde{A}^{-1}A)}[r](\widetilde{A}^{-1}x). \end{aligned}$$

The derivation from the third line to the fourth line is due to the orthogonality of $\widetilde{A}$. Thus for any element x_i in x, we have

$$\frac{\partial}{\partial x_i^{(A)}}\left[\pi^R_{\widetilde{A}}[r]\right](x) = \frac{\partial}{\partial x_i^{(\widetilde{A}^{-1}A)}}[r](\widetilde{A}^{-1}x).$$

Furthermore,

$$\begin{aligned} &\nabla^{(A)}\left[\frac{\partial}{\partial x_i^{(A)}}\left[\pi^R_{\widetilde{A}}[r]\right]\right](x) \\ =&A^{-1}\nabla\left[\frac{\partial}{\partial x_i^{(\widetilde{A}^{-1}A)}}[r](\widetilde{A}^{-1}x)\right] \\ =&(\widetilde{A}^{-1}A)^{-1}\nabla\left[\frac{\partial}{\partial x_i^{(\widetilde{A}^{-1}A)}}[r]\right](\widetilde{A}^{-1}x) \\ =&\nabla^{(\widetilde{A}^{-1}A)}\left[\frac{\partial}{\partial x_i^{(\widetilde{A}^{-1}A)}}[r]\right](\widetilde{A}^{-1}x). \end{aligned}$$

Then we have that for any elements x_i and x_j in x,

$$\frac{\partial}{\partial x_i^{(A)}}\frac{\partial}{\partial x_j^{(A)}}\left[\pi_{\widetilde{A}}^{R}[r]\right](x)=\frac{\partial}{\partial x_i^{(\tilde{A}^{-1}A)}}\frac{\partial}{\partial x_j^{(\tilde{A}^{-1}A)}}[r](\tilde{A}^{-1}x).$$

In this way, it is easy to prove that (12) is satisfied for all the differential operator terms in $\chi^{(\cdot)}$. Finally, as $\chi^{(\cdot)}$ is a linear combination of above terms, (12) is satisfied. Easily, (10) is satisfied.

As for (11), similarly, $\forall x\in\mathbb{R}^n, A\in O(n)$,

$$\begin{aligned}\Phi\left[\pi_{\widetilde{A}}^{E}[e]\right](x,A)&=\Phi\left[e(\widetilde{A}^{-1}x,\widetilde{A}^{-1}A)\right]\\&=\int_{O(n)}\chi_B^{(A)}\left[e(\widetilde{A}^{-1}x,\widetilde{A}^{-1}AB)\right]d\nu(B)\\&=\int_{O(n)}\chi_B^{(A)}\left[\pi_{\widetilde{A}}^{R}[e](x,\widetilde{A}^{-1}AB)\right]d\nu(B)\\&=\int_{O(n)}\chi_B^{(\widetilde{A}^{-1}A)}[e](\widetilde{A}^{-1}x,\widetilde{A}^{-1}AB)d\nu(B)\\&=\pi_{\widetilde{A}}^{E}\left[\int_{O(n)}\chi_B^{(A)}[e](x,AB)d\nu(B)\right]\\&=\pi_{\widetilde{A}}^{E}[\Psi[e]](x,A).\end{aligned}$$

The derivation from the third line to the fourth line is due to (12). So (11) is satisfied.

Furthermore, as differential operators are naturally translation-equivariant, it is easy to verify that Ψ and Φ are also equivariant over $E(n)$. Consequently, according to the working spaces, we set a Ψ as the first layer, followed by multiple Φ's, inserted by pointwise nonlinearities, e.g., ReLUs, that do not disturb the equivariance. Finally, we can get a system where equivariance can be preserved across multiple layers.

3.2.2. Under Subgroup of Orthogonal Transformation

The above theorem can be easily extended to subgroups of $E(n)$. Here we consider a subgroup $\tilde{E}(n)$ with the form $\mathbb{R}^n\rtimes S$, where S is a subgroup of $O(n)$. Similarly, we denote the smooth feature map defined on $\tilde{E}(n)$ as $\tilde{e}$ and the function space as $C^\infty(\tilde{E}(n))$.

The definition of the differential operator Ψ^S is the similar with (8):

$$\forall(x,A)\in\tilde{E}(n),\quad \Psi^S[r](x,A)=\chi^{(A)}[r](x), \tag{13}$$

where the only difference is that $A\in S$. If S is a discrete group, the differential operator Φ^S is:

$$\forall(x,A)\in\tilde{E}(n),\quad \Phi^S[\tilde{e}](x,A)=\sum_{B\in S}\chi_B^{(A)}[\tilde{e}](x,AB), \tag{14}$$

where $A\in S$. Following (2) and (4), we can define $\pi_{\widetilde{A}}^{R}$ and $\pi_{\widetilde{A}}^{\tilde{E}}$, where $\tilde{A}\in S$. We can get the similar result:

$$\Psi^S\left[\pi_{\widetilde{A}}^{R}[r]\right]=\pi_{\widetilde{A}}^{\tilde{E}}\left[\Psi^S[r]\right], \tag{15}$$

$$\Phi^S\left[\pi_{\widetilde{A}}^{\tilde{E}}[\tilde{e}]\right]=\pi_{\widetilde{A}}^{\tilde{E}}\left[\Phi^S[\tilde{e}]\right]. \tag{16}$$

Easily, they are also equivariant w.r.t. $\tilde{E}(n)$.

4. PDO-eConvs

In this section, we apply our theory to 2D digital images, and derive approximately equivariant convolutions in the discrete domain. As they are designed using PDOs, we refer to them as PDO-eConvs. To begin with, we show how to apply PDOs on discrete images and feature maps with convolutional filters, respectively.

4.1. Differential Operators Acting on Discrete Features

We can view discrete digital images as samples from smooth functions defined on the 2D plane. Formally, we assume that an image data $\boldsymbol{I}\in\mathbb{R}^{n\times n}$ represents a two-dimensional grid function obtained by discretizing a smooth function $r:[0,1]\times[0,1]:\to\mathbb{R}$ at the cell-centers of a regular grid with $n\times n$ cells and a mesh size $h=1/n$, i.e., for $i,j=1,2,\ldots,n$,

$$\boldsymbol{I}_{i,j}=r(x_i,y_j),$$

where $x_i=(i-\frac{1}{2})h$ and $y_j=(j-\frac{1}{2})h$.

Accordingly, intermediate feature maps in CNNs are multi-channel matrices. Similarly, it can be seen as the discretizations of continuous functions defined on $\tilde{E}$, where $\tilde{E}=\mathbb{R}^2\rtimes S$ and S is a subgroup of $O(2)$. Formally, a feature map $\boldsymbol{F}$ represents a three-dimensional grid function sampled from a smooth function $e:[0,1]^2\times S:\to\mathbb{R}$. For $i,j=1,2,\ldots n$,

$$\boldsymbol{F}_{i,j}^k=e(x_i,y_j,k), \tag{17}$$

where $x_i=(i-\frac{1}{2})h, y_j=(j-\frac{1}{2})h$ and $k\in S$ which represents its channel index. Here, for ease of presentation, we only consider that inputs and intermediate feature maps are all single-valued functions, and the theory can be easily extended to multi-valued functions.

With the understanding that features are sampled from continuous functions, we can implement differential operations on features. Particularly, we use convolutions to approximate differential operations, which have been widely used in image processing. For example, the operator $\frac{\partial}{\partial x}$ acting on images and feature maps can be approximated by the following 3×3 convolutional filter with quadratic precision:

$$\frac{\partial}{\partial x}[r](x_i, y_j) = \left(\frac{1}{2h} \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} * \boldsymbol{I} \right)_{i,j} + O(h^2),$$

$$\frac{\partial}{\partial x}[e](x_i, y_j, k) = \left(\frac{1}{2h} \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} * \boldsymbol{F}^k \right)_{i,j} + O(h^2),$$

where $*$ denotes the convolution operation.

4.2. From Group Equivariant Differential Operators to PDO-eConvs

Firstly, we choose the polynomial H from the connection between differential operators and convolutions. Ruthotto & Haber (2018) showed that we can relate a 3×3 convolutional filter to a differential operator, $\mathcal{D}$, which is a linear combination of 9 linearly independent PDOs[4].

$$\begin{aligned} \mathcal{D} =& \beta_1 \partial_0 + \beta_2 \partial_x + \beta_3 \partial_y + \beta_4 \partial_{xx} + \beta_5 \partial_{xy} \\ &+ \beta_6 \partial_{yy} + \beta_7 \partial_{xxy} + \beta_8 \partial_{xyy} + \beta_9 \partial_{xxyy}. \end{aligned} \tag{18}$$

In addition, we observe that all differential operators in (19) can be approximated using 3×3 convolutional filters (see Supplementary Material 1.1) with quadratic precision. It is to say that we can always approximate the differential operators defined in (19) using a 3×3 filter with quadratic precision. For this reason, we choose

$$\begin{aligned} H(u, v; \boldsymbol{\beta}) =& \beta_1 + \beta_2 u + \beta_3 v + \beta_4 u^2 + \beta_5 uv \\ &+ \beta_6 v^2 + \beta_7 u^2 v + \beta_8 uv^2 + \beta_9 u^2 v^2. \end{aligned} \tag{19}$$

In this way, $\mathcal{D}$ equals $\chi^{(I)}$, which is also the canonical differential operator of $\chi^{(A)}$'s, indexed by the identity matrix. Using the transformation in (6), we can calculate all the expressions of $\chi^{(A)}$'s easily. Particularly, these transformed differential operators share the same parameters $\boldsymbol{\beta}$, indicating greater parameter efficiency.

In computation, we observe that some new partial derivatives, e.g., $\partial_{xxx}, \partial_{xxxx}$, may occur in some $\chi^{(A)}$'s, where $A \in S$. Fortunately, the orders of these new partial derivatives are all below five, and we can use the filters with the size of 5×5 (see Supplementary Material 1.2) to approximate them with quadratic precision.

Now we investigate the group we use. According to (9) and (14), if S is a continuous group, we need to conduct integration. However, for the computation issue, it seems impossible to consider all the orthogonal transformations in $O(2)$. So we consider S to be a discrete subgroup of $O(2)$. Still, our theory is satisfied for feature maps defined on $\tilde{E}$ (see Section 3.2.2). Particularly, noting that $O(2)$ is generated by reflections and rotations, we set the subgroup S to be generated by reflections and rotations by $2\pi/n$. As a result, $\tilde{E} = pnm$. If without reflections, $\tilde{E} = pn$. Discrete groups pnm and pn have been introduced in Section 1.

[4]For ease of presentation, we denote the identity operator as ∂_0, and view it as a special PDO.

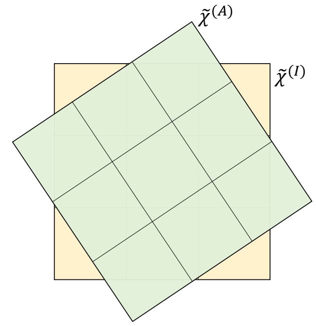

Figure 3. The canonical convolutional filter $\tilde{\chi}^{(I)}$ and its rotated version $\tilde{\chi}^{(A)}$.

Finally, we discretize the equivariant differential operator Ψ with corresponding convolutional filters. As a result, we can get a new operator, $\tilde{\Psi}$, which is actually a set of convolution operators indexed by A:

$$\forall A \in S, \quad \tilde{\chi}^{(A)} = \sum_{i \in \Gamma} C_i^{(A)} \tilde{u}_i, \tag{20}$$

where Γ indexes all the filters we use, $C_i^{(A)}$ are derived by substituting (6) into (5) and $\tilde{u}_i$ is the convolutional filter related to the PDO ∂_i (e.g., $\tilde{u}_0$ and $\tilde{u}_{xy}$ are related to ∂_0 and ∂_{xy}, respectively), then

$$(\widetilde{\Psi} * \boldsymbol{I})^A = \tilde{\chi}^{(A)} * \boldsymbol{I}. \tag{21}$$

Similarly, we can get a new convolution operator $\tilde{\Phi}$ by discretizing (14). Without ambiguity, we also use $*$ to denote the corresponding convolution operation. To be specific,

$$\forall A \in S, \quad \left(\tilde{\Phi} * \boldsymbol{F}\right)^A = \sum_{k \in S} \tilde{\chi}_k^{(A)} * \boldsymbol{F}^{Ak}, \tag{22}$$

where Ak is a group product on the group S, which respresents the channel index of $\boldsymbol{F}$, and $\boldsymbol{F}^{Ak} \in \mathbb{R}^{n \times n}$.

We refer to $\tilde{\Psi}$ and $\tilde{\Phi}$ as PDO-eConvs, because they are equivariant convolutions based on PDOs. Following (Cohen & Welling, 2016), we replace all the conventional convolutions in an existing CNN with our PDO-eConvs, and get the corresponding group equivariant CNN w.r.t. $\tilde{E}$.

Let us have a more detailed look at (20). Some convolutional filters like u_{xxxx} are of size 5×5, thus for some $A \in \tilde{E}$, $\tilde{\chi}^{(A)}$ is also of size 5×5, while the canonical convolutional filter $\tilde{\chi}^{(I)}$ is of size 3×3. We can explain the phenomenon in this way. By definition, the differential operator $\chi^{(A)}$ is transformed from $\chi^{(I)}$. Intuitively, we can also view the convolutional filter $\tilde{\chi}^{(A)}$ as a transformed version of $\tilde{\chi}^{(I)}$. We assume the transformation to be the rotation. As shown in Figure 3, $\tilde{\chi}^{(A)}$ is a rotated version of $\tilde{\chi}^{(I)}$, which overflows the original 3×3 area. So it makes sense to use a larger filter to represent some transformed filters. That 5×5 is sufficient is because the rotated 3×3 mask can always be covered by a 5×5 square, noting that $5\geq 3\sqrt{2}$.

4.3. Approximation Error of Equivariance

When we discretize the differential operators Ψ and Φ, errors occur, leading to equivariance disturbance. Nonetheless, we can still achieve approximate equivariance. Here, we analyze the approximation error of our PDO-eConvs.

Theorem 2 $\forall \tilde{A}\in S$,

$$\tilde{\Psi} * \pi^{R}_{\tilde{A}}[\boldsymbol{I}] = \pi^{\tilde{E}}_{\tilde{A}}\left[\tilde{\Psi} * \boldsymbol{I}\right] + O(h^2), \tag{23}$$

$$\tilde{\Phi} * \pi^{\tilde{E}}_{\tilde{A}}[\boldsymbol{F}] = \pi^{\tilde{E}}_{\tilde{A}}\left[\tilde{\Phi} * \boldsymbol{F}\right] + O(h^2), \tag{24}$$

where transformations such as rotations or mirror reflections acting on images are defined as $(\pi^{R}_{\tilde{A}}[\boldsymbol{I}])_{i,j} = (\pi^{R}_{\tilde{A}}[r])(x_i,y_j)$ *and transformations acting on feature maps are* $(\pi^{\tilde{E}}_{\tilde{A}}[\boldsymbol{F}])^k_{i,j} = (\pi^{\tilde{E}}_{\tilde{A}}[e])(x_i,y_j,k)$.

Proof 2 $\forall A\in S$, *the operator* $\chi^{(A)}$ *is a linear combination of differential operators and* $\tilde{\chi}^{(A)}$ *is a combination of corresponding convolution operators. Hence if* r *is a smooth function,*

$$\chi^{(A)}[r](x_i,y_j) = \left(\tilde{\chi}^{(A)} * \boldsymbol{I}\right)_{i,j} + O(h^2),$$

$$\chi^{(A)}\left[\pi^{R}_{\tilde{A}}[r]\right](x_i,y_j) = \left(\tilde{\chi}^{(A)} * \pi^{R}_{\tilde{A}}[\boldsymbol{I}]\right)_{i,j} + O(h^2),$$

i.e.,

$$\Psi[r](x_i,y_j,A) = \left(\tilde{\Psi} * \boldsymbol{I}\right)^{A}_{i,j} + O(h^2),$$

$$\Psi\left[\pi^{R}_{\tilde{A}}[r]\right](x_i,y_j,A) = \left(\tilde{\Psi} * \pi^{R}_{\tilde{A}}[\boldsymbol{I}]\right)^{A}_{i,j} + O(h^2). \tag{25}$$

Easily, we have

$$\pi^{\tilde{E}}_{\tilde{A}}\left[\Psi[r]\right](x_i,y_j,A) = \left(\pi^{\tilde{E}}_{\tilde{A}}\left[\tilde{\Psi} * \boldsymbol{I}\right]\right)^{A}_{i,j} + O(h^2). \tag{26}$$

From (10) we know that the left hand sides of (25) and (26) equal, hence the right hand sides of the two equation are the same, which results in (23). We can prove (24) analogously.

4.4. Weight Initialization Scheme

An important practical issue in the training phase is an appropriate initialization of weights. When the variances of weights are chosen too high or too low, the signals propagating through the network are amplified or suppressed exponentially with depth. Glorot & Bengio (2010) and He et al. (2015) investigated this problem and proposed widely used initialization schemes. However, our filters are not parameterized in a pixel basis but as linear combinations of several PDOs, thus the above-mentioned initialization schemes cannot directly be adopted for our PDO-eConvs.

To be specific, we consider the canonical filter $\tilde{\chi}^{(I)}$ in each PDO-eConv, and initialize it with He's initialization scheme (He et al., 2015). Then we initialize the parameters $\boldsymbol{\beta}$ of the PDO-eConv by solving the linear equation

$$\begin{aligned}\tilde{\chi}^{(I)} =& \beta_1\tilde{u}_0 + \beta_2\tilde{u}_x + \beta_3\tilde{u}_y + \beta_4\tilde{u}_{xx} + \beta_5\tilde{u}_{xy} \\ &+ \beta_6\tilde{u}_{yy} + \beta_7\tilde{u}_{xxy} + \beta_8\tilde{u}_{xyy} + \beta_9\tilde{u}_{xxyy}.\end{aligned} \tag{27}$$

with the initialized $\tilde{\chi}^{(I)}$. In this way, the canonical filter is initialized with He's initialization scheme. Since other filters are obtained by transforming the canonical filters, they also have appropriate variances. We initialize each $\tilde{\Psi}_k$ in (22) in the same way. We use this method to initialize all the PDO-eConvs in experiments and all the experiments are implemented using Tensorflow.

5. Experiments

5.1. Rotated MNIST

The most commonly used dataset for validating rotation-equivariant algorithms is MNIST-rot-12k (Larochelle et al., 2007). It contains the handwritten digits of the classical MNIST, rotated by a random angle from 0 to 2π (full angle). This dataset contains 12,000 training images and 50,000 test images, respectively. We randomly select 2,000 training images as a validation set. We choose the model with the lowest validation error during training. For preprocessing, we normalize the images using the channel means and standard deviations.

Without Data Augmentation Firstly, we evaluate the performance of PDO-eConvs on MNIST-rot-12k without data augmentation via the CNN architecture used in (Cohen & Welling, 2016). It contains 6 layers of 3×3 convolutions, 20 channels in each layer, ReLU functions, batch normalization (Ioffe & Szegedy, 2015), and max pooling after layer 2.

We consider the group $p8$ and replace each convolution by a $p8$-convolution, divided the number of filters by $\sqrt{8}$, in order to keep the numbers of parameters nearly the same. Thus we use 7 filters on each layer. Particularly, batch normalization should be implemented with a single scale and a single bias per PDO-eConv map to preserve equivariance.

The model is trained using the Adam algorithm (Kingma & Ba, 2015) with a weight decay of 0.01. We use the weight

Table 1. Error rates on MNIST-rot-12k without data augmentation.

Network	Test Error (%)	params
ScatNet-2 (Bruna & Mallat, 2013)	7.48	-
PCANet-2 (Chan et al., 2015)	7.37	-
TIRBM (Sohn & Lee, 2012)	4.2	-
ORN-8 (ORNAlign) (Zhou et al., 2017)	2.25	0.53M
TI-Pooling (Laptev et al., 2016)	2.2	13.3M
CNN	5.03	22k
G-CNN (Cohen & Welling, 2016)	2.28	25k
PDO-eConv (ours)	**1.87**	26k

initialization method introduced in Section 4.4 for PDO-eConvs and Xavier initialization (Glorot & Bengio, 2010) for the fully connected layer. We train using batch size 128 for 200 epochs. The initial learning rate is set to 0.001 and is divided by 10 at 50% and 75% of the total number of training epochs. We set the dropout rate as 0.2.

As shown in Table 1, with comparable numbers of parameters, our proposed PDO-eConv achieves 1.87% test error, outperforming conventional CNN (5.03%) and G-CNN (2.28%), which is equivariant on group $p4$. This is mainly because that our model is rotation-equivariant w.r.t. smaller rotation angles, which brings in better generalization. ORN-8 also deals with an 8-fold rotational symmetry and adopts an extra strategy, ORNAlign, to refine feature maps. Compared with ORN-8 (ORNAlign), our method still results in lower test error, using far fewer numbers of parameters (26k vs. 0.53M). TI-Pooling is a representative model of transformation-invariant CNNs, which use parallel siamese architectures. Compared with it, PDO-eConv performs better (1.87% vs. 2.2%) using far fewer parameters (26k vs. 13.3M) and has much lower computational complexity.

Competitive Result with Data Augmentation We compare the performance of our PDO-eConv with some more competitive models, using data augmentation and a larger model with 7 layers. These layers have 16, 16, 32, 32, 32, 64 and 64 output channels, respectively. We use a spatial pooling and an orientation pooling after the final PDO-eConv layer, in order to get rotation-invariant features. Following (Weiler et al., 2018), we augment the dataset with continuous rotations during training time. This model is trained using stochastic gradient descent (SGD) and a Nesterov momentum (Sutskever et al., 2013) of 0.9 without dampening. We train this model for 300 epochs, starting with a learning rate of 10^{-2} and reducing it gradually to 10^{-5}.

Table 2. Competitive results on MNIST-rot-12k.

Method	Test Error (%)
H-Net (Worrall et al., 2017)	1.69
OR-TIPooling (Zhou et al., 2017)	1.54
RotEqNet (Marcos et al., 2017)	1.09
PTN-CNN (Esteves et al., 2018)	0.89
E2CNN (Cesa & Weiler, 2019)	0.716
SFCNN (Weiler et al., 2018)	0.714
PDO-eConv (ours)	**0.709**

As shown in Table 2, E2CNN and SFCNN achieve 0.716% and 0.714% test error on rotated MNIST, respectively. Compared with SFCNN, our method achieves a comparable result, 0.709% test error, using only 10% parameters. To be specific, our method uses 0.65M parameters, while SFCNN needs 6.5M parameters. Also, SFCNN used a much larger architecture and larger kernel sizes (7×7 and 9×9), which relate to a much larger computational cost. E2CNN replicates the architecture used in SFCNN, so it also relates to a huge computational cost.

5.2. Natural Image Classfication

Although most objects in natural scene images are up-right, rotations could exist in small scales. Besides, equivariance to a transformation group brings in more parameter sharing, which may improve the parameter efficiency. Here we evaluate the performance of our PDO-eConvs on two common natural image datasets, CIFAR-10 (C10) and CIFAR-100 (C100) (Krizhevsky & Hinton, 2009), respectively.

The two CIFAR datasets consist of colored natural images with 32×32 pixels. C10 consists of images drawn from 10 classes and C100 from 100. The training and the test sets contain 50,000 and 10,000 images, respectively. We randomly select 5,000 training images as a validation set. We choose the model with the lowest validation error during training. We adopt a standard data augmentation scheme (mirroring/shifting) (Lee et al., 2015) that is widely used for these two datasets. For preprocessing, we normalize the images using the channel means and standard deviations.

To evaluate our method, we take ResNet (He et al., 2016) as the basic model, which consists of an initial convolution layer, followed by three stages of $2n$ convolution layers using k_i filters at stage i, followed by a final classification layer ($6n+2$ layers in total). We replace all convolution layers of ResNets by our PDO-eConvs and implement batch normalization with a single scale and a single bias per PDO-eConv map. Also, we scale the number of filters to keep the numbers of parameters approximately the same. All the models are trained using SGD and a Nesterov momentum (Sutskever et al., 2013) of 0.9 without dampening. We train using batch size 128 for 300 epochs, weight decay of 0.001. The initial learning rate is set to 0.1 and is divided by 10 at 50% and 75% of the total number of training epochs. Similarly, we use the weight initialization method introduced in Section 4.4 for our PDO-eConvs and Xavier initialization for the fully connected layer. We report the results of our methods in Table 3.

Table 3. Results on the natural image classification benchmark. In the second column, G is the group where equivariance can be preserved.

Method	G	Depth	C10	C100	params
ResNet (He et al., 2016)	$\mathbb{Z}^2$	26	11.5	31.66	0.37M
HexaConv (Hoogeboom et al., 2018)	$p6$	26	9.98	-	0.34M
	$p6m$	26	8.64	-	0.34M
PDO-eConv (ours)	$p6$	26	5.65	27.13	0.36M
	$p6m$	26	5.38	27.00	0.37M
ResNet	$\mathbb{Z}^2$	44	5.61	24.08	2.64M
G-CNN (Cohen & Welling, 2016)	$p4m$	44	4.94	23.19	2.62M
PDO-eConv (ours)	$p8$	44	3.68	20.01	2.62M
ResNet	$\mathbb{Z}^2$	1001	4.92	22.71	10.3M
Wide ResNet (Zagoruyko & Komodakis, 2016)	$\mathbb{Z}^2$	26	4.00	19.25	36.5M
G-CNN (Cohen & Welling, 2016)	$p4m$	26	4.17	-	7.2M
PDO-eConv (ours)	$p8$	26	**3.50**	**18.40**	4.6M

Following HexaConv, we use our PDO-eConvs to establish models that are equivariant to group $p6$ ($p6m$), where $n = 4$ and $k_i = 6, 13, 26$ ($k_i = 6, 9, 18$). Using comparable numbers of parameters, our methods perform significantly better than HexaConv (5.38% vs. 8.64% on C10). In addition, HexaConvs require extra memory to store hexagonal images while our PDO-eConvs do not need so.

We evaluate PDO-eConvs using ResNet-44, where $n = 7$ and $k_i = 11, 23, 45$. Compared with G-CNNs, our PDO-eConvs achieve significantly better performance using comparable numbers of parameters (3.68% vs. 4.94% on C10, and 20.01% vs. 23.19% on C100). When evaluated on ResNet-26, where $n = 4, k_i = 20, 40, 80$, PDO-eConv results in 3.50% test error, much better than 4.17% resulted from G-CNN, yet using much fewer parameters (4.6M vs. 7.2M). This is mainly because that PDO-eConvs can deal with an 8-fold rotational symmetry, which exploit more rotational symmetries compared with G-CNN.

Finally, we compare our models with deeper ResNets (ResNet-1001) and wider ResNets (Wide ResNet). As shown in Table 3, PDO-eConvs perform betterr (3.50% vs. 4.00% in C10 and 18.40% vs. 19.25% in C100) using only 12.6% parameters (4.6M vs. 36.5M). Particularly, PDO-eConvs can also be viewed as introducing a weight sharing scheme across channels, and the results indicate that our method can not only save parameters, but also improve the performance remarkably.

6. Conclusion

We utilize PDOs to design a system which is exactly equivariant to a much more general continuous group, the n-dimension Euclidean group. We use numerical schemes to implement these PDOs and derive approximately equivariant convolutions, PDO-eConvs. Particularly, we provide an error analysis and show that the approximation error is of the quadratic order. Extensive experiments verify the effectiveness of our method.

In this work, we only conduct experiments on 2D images. Actually, our theory can deal with the data with any dimension. We will explore more possibilities in the future.

Acknowledgements

This work was supported by the National Key Research and Development Program of China under grant 2018AAA0100205. Z. Lin is supported by NSF China (grant no.s 61625301 and 61731018), Major Scientific Research Project of Zhejiang Lab (grant no.s 2019KB0AC01 and 2019KB0AB02), Beijing Academy of Artificial Intelligence, and Qualcomm.

References

Bruna, J. and Mallat, S. Invariant scattering convolution networks. *TPAMI*, 35(8):1872–1886, 2013.

Cesa, G. and Weiler, M. General E(2)-equivariant steerable CNNs. In *NeurIPS*, pp. 14334–14345, 2019.

Chan, T.-H., Jia, K., Gao, S., Lu, J., Zeng, Z., and Ma, Y. PCANet: A simple deep learning baseline for image classification? *TIP*, 24(12):5017–5032, 2015.

Cohen, T. and Welling, M. Group equivariant convolutional networks. In *ICML*, pp. 2990–2999, 2016.

Cohen, T. S. and Welling, M. Steerable CNNs. In *ICLR*, 2017.

Dong, B., Jiang, Q., and Shen, Z. Image restoration: Wavelet frame shrinkage, nonlinear evolution pdes, and beyond. *Multiscale Modeling & Simulation*, 15(1):606–660, 2017.

Esteves, C., Allenblanchette, C., Zhou, X., and Daniilidis, K. Polar transformer networks. In *ICLR*, 2018.

Fang, C., Zhao, Z., Zhou, P., and Lin, Z. Feature learning via partial differential equation with applications to face recognition. *Pattern Recognition*, 69:14–25, 2017.

Glorot, X. and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, pp. 249–256, 2010.

He, K., Zhang, X., Ren, S., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, pp. 1026–1034, 2015.

He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. In *ECCV*, pp. 630–645. Springer, 2016.

Helor, Y. and Teo, P. C. Canonical decomposition of steerable functions. *Journal of Mathematical Imaging and Vision*, 9(1):83–95, 1996.

Hinton, G. E., Sabour, S., and Frosst, N. Matrix capsules with EM routing. In *ICLR*, 2018.

Hoogeboom, E., Peters, J. W., Cohen, T. S., and Welling, M. HexaConv. In *ICLR*, 2018.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pp. 448–456, 2015.

Jaderberg, M., Simonyan, K., Zisserman, A., and Kavukcuoglu, K. Spatial transformer networks. In *NeurIPS*, pp. 2017–2025, 2015.

Jain, A. K. and Jain, J. Partial differential equations and finite difference methods in image processing–Part II: Image restoration. *IEEE Transactions on Automatic Control*, 23(5):817–834, 1978.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Koenderink, J. J. The structure of images. *Biological Cybernetics*, 50(5):363–370, 1984.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, pp. 1097–1105, 2012.

Laptev, D., Savinov, N., Buhmann, J. M., and Pollefeys, M. TI-POOLING: transformation-invariant pooling for feature learning in convolutional neural networks. In *CVPR*, pp. 289–297, 2016.

Larochelle, H., Erhan, D., Courville, A., Bergstra, J., and Bengio, Y. An empirical evaluation of deep architectures on problems with many factors of variation. In *ICML*, pp. 473–480, 2007.

Lee, C.-Y., Xie, S., Gallagher, P., Zhang, Z., and Tu, Z. Deeply-supervised nets. In *AISTATS*, pp. 562–570, 2015.

Lenc, K. and Vedaldi, A. Understanding image representations by measuring their equivariance and equivalence. In *CVPR*, pp. 991–999, 2015.

Liu, R., Lin, Z., Zhang, W., Tang, K., and Su, Z. Toward designing intelligent PDEs for computer vision: An optimal control approach. *Image and vision computing*, 31 (1):43–56, 2013.

Long, Z., Lu, Y., Ma, X., and Dong, B. PDE-Net: Learning PDEs from data. In *ICML*, pp. 5067–5078. International Machine Learning Society (IMLS), 2018.

Long, Z., Lu, Y., and Dong, B. PDE-Net 2.0: Learning PDEs from data with a numeric-symbolic hybrid deep network. *Journal of Computational Physics*, 399:108925, 2019.

Marcos, D., Volpi, M., Komodakis, N., and Tuia, D. Rotation equivariant vector field networks. In *ICCV*, pp. 5048–5057, 2017.

Osher, S. and Rudin, L. Feature-oriented image enhancement using shock filters. *SIAM Journal on Numerical Analysis*, 27(4):919–940, 1990.

Perona, P. and Malik, J. Scale-space and edge detection using anisotropic diffusion. *IEEE TPAMI*, 12(7):629–639, 1990.

Ruthotto, L. and Haber, E. Deep neural networks motivated by partial differential equations. *Journal of Mathematical Imaging and Vision*, pp. 1–13, 2018.

Sabour, S., Frosst, N., and Hinton, G. E. Dynamic routing between capsules. In *NeurIPS*, pp. 3856–3866, 2017.

Sohn, K. and Lee, H. Learning invariant representations with local transformations. In *ICML*, pp. 1339–1346, 2012.

Sutskever, I., Martens, J., Dahl, G., and Hinton, G. On the importance of initialization and momentum in deep learning. In *ICML*, pp. 1139–1147, 2013.

Weiler, M., Hamprecht, F. A., and Storath, M. Learning steerable filters for rotation equivariant CNNs. In *CVPR*, pp. 849–858, 2018.

Witkin, A. Scale-space filtering. In *IJCAI*, pp. 329–332, 1987.

Worrall, D. E., Garbin, S. J., Turmukhambetov, D., and Brostow, G. J. Harmonic networks: Deep translation and rotation equivariance. In *CVPR*, pp. 5028–5037, 2017.

Zagoruyko, S. and Komodakis, N. Wide residual networks. In *BMVC*, 2016.

Zhou, Y., Ye, Q., Qiu, Q., and Jiao, J. Oriented response networks. In *CVPR*, pp. 519–528, 2017.