

SUPPLEMENTARY MATERIAL: NEURAL EPDOs: SPATIALLY ADAPTIVE EQUIVARIANT PARTIAL DIFFERENTIAL OPERATOR BASED NETWORKS

A GROUP REPRESENTATION

Formally, a group representation ρ of group G is a group homomorphism: $G \rightarrow GL(V)$, i.e., $\forall g_1, g_2 \in G, \rho(g_1 g_2) = \rho(g_1) \rho(g_2)$. Here, V is a linear space, and $GL(V)$ is the general linear group on V (the set of all bijective linear transformations $V \rightarrow V$, usually adopted as matrices). Here, we mainly introduce three types of group representations that are often adopted in equivariant researches.

Regular Representation It is a group representation commonly used for discrete groups. It acts on a vector space $\mathbb{R}^{|G|}$ by permuting its axes. If we associate every axis with a group element, then the action $\rho_{reg}^G(g)$ of regular representation permutes the axis associated with g_1 to the axis associated with gg_1 . Specifically, the matrix form of $\rho_{reg}^G(g)$ is $\rho_{reg}^G(g)_{g_i, g_j} = \delta_{g_i, g_j g}^{-1}$, where $g_i, g_j, g_j g \in G$ denote the corresponding axes. Feature maps with regular representations as transformation type are called regular fields.

Quotient Representation It is very similar to regular representation. The vector space is $\mathbb{R}^{|G|/|H|}$ and each axis is associated by cosets gH in the quotient space G/H of H , where H is a subgroup of G . The action $\rho_{quot}^{G/H}(g)$ of quotient representation permutes the axis associated with $g_1 H$ to the one associated with $gg_1 H$. Under the canonical basis, $\rho_{quot}^{G/H}(g)$ is of the matrix form $\rho_{quot}^{G/H}(g)_{g_1 H, g_2 H} = \delta_{g_1 H, g g_2 H}$. Feature maps with quotient representations as transformation type are called quotient fields.

Irreducible Representation of $SO(2)$ It refers to representations which cannot be decomposed into more basic representations. One dimensional irreducible representation of $SO(2)$ is trivial representation, denoted as ψ_0 . Two-dimensional irreducible representations of $SO(2)$ are

$$\psi_k(\theta) = \begin{bmatrix} \cos(k\theta), & -\sin(k\theta) \\ \sin(k\theta), & \cos(k\theta) \end{bmatrix}, \quad k \in \mathbb{N}^+, \quad (16)$$

where $\theta \in [0, 2\pi]$ denotes a rotation.

B PROOF OF LEMMA.1

Lemma 1 $\hat{\rho}(g)$ defined in Eq.(5) is a group representation of g on $\mathbb{R}^{|\Gamma_N|}$.

Proof 1 As the elementary PDOs would not change when translation transform acts on \mathbf{f} , we consider $\forall g_1, g_2 \in G$ act on \mathbf{f} sequentially, and let $\tilde{\mathbf{f}}(x) := \mathbf{f}(g_1^{-1}x)$, we have:

¹ $\delta_{i,j} = 1$ if $i = j$, and $\delta_{i,j} = 0$ otherwise.

$$\forall \alpha \in \Gamma_N, \partial^\alpha [\tilde{\mathbf{f}}(g_2^{-1}x)](x) = \sum_{\beta \in \Gamma_N} \hat{\rho}_{\alpha,\beta}(g_2) \partial^\beta [\tilde{\mathbf{f}}](g_2^{-1}x) \quad (17)$$

$$= \sum_{\beta \in \Gamma_N} \hat{\rho}_{\alpha,\beta}(g_2) \sum_{\gamma \in \Gamma_N} \hat{\rho}_{\beta,\gamma}(g_1) \partial^\gamma [\mathbf{f}](g_1^{-1}g_2^{-1}x) \quad (18)$$

$$= \sum_{\gamma \in \Gamma_N} \left(\sum_{\beta \in \Gamma_N} \hat{\rho}_{\alpha,\beta}(g_2) \hat{\rho}_{\beta,\gamma}(g_1) \right) \partial^\gamma [\mathbf{f}]((g_2g_1)^{-1}x) \quad (19)$$

$$= \sum_{\gamma \in \Gamma_N} (\hat{\rho}(g_2)\hat{\rho}(g_1))_{\alpha,\gamma} \partial^\gamma [\mathbf{f}]((g_2g_1)^{-1}x) \quad (20)$$

On the other hand,

$$\forall \alpha \in \Gamma_N, \partial^\alpha [\tilde{\mathbf{f}}(g_2^{-1}x)](x) = \partial^\alpha [\mathbf{f}((g_2g_1)^{-1}x)](x) = \sum_{\gamma \in \Gamma_N} \hat{\rho}_{\alpha,\gamma}(g_2g_1) \partial^\gamma [\mathbf{f}]((g_2g_1)^{-1}x) \quad (21)$$

We can get $\forall g_1, g_2 \in G, \hat{\rho}(g_2g_1) = \hat{\rho}(g_2) \hat{\rho}(g_1)$. Hence $\hat{\rho}$ is a group representation of group G .

C COMPUTATION OF $\hat{\rho}(g)$

It is not difficult to find that elementary PDOs of different orders are decoupled under coordinate transformation, *i.e.*,

$$\hat{\rho}_{\alpha,\beta}(g) = 0 \quad \text{if} \quad \alpha_1 + \alpha_2 \neq \beta_1 + \beta_2 \quad (22)$$

where $\alpha = (\alpha_1, \alpha_2), \beta = (\beta_1, \beta_2)$. In order to compute all $\hat{\rho}_{\alpha,\beta}(g)$ in which $\alpha_1 + \alpha_2 = \beta_1 + \beta_2 = s$, we give a following program:

Algorithm 1 Computation of $\hat{\rho}(g)$

Input: Highest order N of considered elementary PDO, group element g

Output: $\hat{\rho}(g)$

```

1:  $\hat{\rho}_{(0,0),(0,0)}(g) \leftarrow 1$ 
2: for  $s = 1$  to  $N$  do
3:   for  $\alpha_1 = 1$  to  $s$  do
4:      $\alpha_2 \leftarrow s - \alpha_1$ 
5:      $\beta_1 \leftarrow s, \beta_2 \leftarrow 0, \hat{\rho}_{\alpha,\beta}(g) \leftarrow g_{1,1} \hat{\rho}_{(\alpha_1-1,\alpha_2),(\beta_1-1,\beta_2)}(g)$ 
6:      $\beta_1 \leftarrow 0, \beta_2 \leftarrow s, \hat{\rho}_{\alpha,\beta}(g) \leftarrow g_{1,2} \hat{\rho}_{(\alpha_1-1,\alpha_2),(\beta_1,\beta_2-1)}(g)$ 
7:     for  $\beta_1 = 1$  to  $s - 1$  do
8:        $\beta_2 = s - \beta_1$ 
9:        $\hat{\rho}_{\alpha,\beta}(g) \leftarrow g_{1,1} \hat{\rho}_{(\alpha_1-1,\alpha_2),(\beta_1-1,\beta_2)}(g) + g_{1,2} \hat{\rho}_{(\alpha_1-1,\alpha_2),(\beta_1,\beta_2-1)}(g)$ 
10:    end for
11:  end for
12:   $\alpha_1 \leftarrow 0, \alpha_2 \leftarrow s$ 
13:   $\beta_1 \leftarrow s, \beta_2 \leftarrow 0, \hat{\rho}_{\alpha,\beta}(g) \leftarrow g_{2,1} \hat{\rho}_{(\alpha_1,\alpha_2-1),(\beta_1-1,\beta_2)}(g)$ 
14:   $\beta_1 \leftarrow 0, \beta_2 \leftarrow s, \hat{\rho}_{\alpha,\beta}(g) \leftarrow g_{2,1} \hat{\rho}_{(\alpha_1,\alpha_2-1),(\beta_1,\beta_2-1)}(g)$ 
15:  for  $\beta_1 = 1$  to  $s - 1$  do
16:     $\beta_2 = s - \beta_1$ 
17:     $\hat{\rho}_{\alpha,\beta}(g) \leftarrow g_{1,1} \hat{\rho}_{(\alpha_1,\alpha_2-1),(\beta_1-1,\beta_2)}(g) + g_{2,1} \hat{\rho}_{(\alpha_1-1,\alpha_2),(\beta_1,\beta_2-1)}(g)$ 
18:  end for
19: end for
```

The above program computes $\hat{\rho}(g)$ recursively and time complexity of the program is proportional to number of $\hat{\rho}_{\alpha,\beta}(g)$ in which $\alpha_1 + \alpha_2 = \beta_1 + \beta_2 = s$.

D PROOF OF PROPOSITION.1

Proposition 1 *The nonlinear PDOs in Eq.(7) are equivariant to affine H if and only if the coefficient generators satisfy the following constraint:*

$$\forall \alpha \in \Gamma_N, \forall g \in G, \forall \mathbf{y} \in \mathbb{R}^{c_{in}}, \quad \sum_{\beta \in \Gamma_N} \hat{\rho}_{\beta, \alpha}(g) \mathbf{W}_{\beta}(\rho(g)\mathbf{y}) \rho(g) = \rho'(g) \mathbf{W}_{\alpha}(\mathbf{y}).$$

Proof 2 *As introduced in Section 3.1, an H -equivariant operator Ψ in Eq.(7) should satisfy the following requirement:*

$$\forall h \in H, \quad \Psi[\pi(h)[\mathbf{f}]] = \pi'(h)[\Psi[\mathbf{f}]], \quad (23)$$

where H is a transformation group, $\pi(h)$ and $\pi'(h)$ are group actions, and \mathbf{f} is the input field. According to the definition of π Eq.(2), the LHS of Eq. (23) becomes:

$$\forall x \in \mathbb{R}^2, \quad \Psi[\pi(h)[\mathbf{f}]](x) = \sum_{\beta \in \Gamma_N} \mathbf{W}_{\beta}(\pi(h)\mathbf{f}(x)) \partial^{\beta}[\pi(h)\mathbf{f}](x) \quad (24)$$

$$= \sum_{\beta \in \Gamma_N} \mathbf{W}_{\beta}(\rho(g)\tilde{\mathbf{f}}(x)) \partial^{\beta}[\rho(g)\tilde{\mathbf{f}}](x), \quad (25)$$

where $\tilde{\mathbf{f}}$ denote $\mathbf{f}(g^{-1}(x - t))$. As the element PDOs are linear operators, we have: $\forall \beta \in \Gamma_N, \partial^{\beta}[\rho(g)\tilde{\mathbf{f}}](x) = \rho(g)\partial^{\beta}[\tilde{\mathbf{f}}](x)$. According to Eq.(5):

$$Eq.(25) = \sum_{\alpha \in \Gamma_N} \sum_{\beta \in \Gamma_N} \hat{\rho}_{\beta, \alpha}(g) \mathbf{W}_{\beta}(\rho(g)\tilde{\mathbf{f}}(x)) \rho(g) \partial^{\alpha}[\mathbf{f}](g^{-1}(x - t)) \quad (26)$$

On the RHS of Eq.(23), we have:

$$\forall x \in \mathbb{R}^2, \quad \pi'(h)[\Psi[\mathbf{f}]] = \pi'(h)[\sum_{\alpha \in \Gamma_N} \mathbf{W}_{\alpha}(\mathbf{f}(x)) \partial^{\alpha}[\mathbf{f}](x)] \quad (27)$$

$$= \sum_{\alpha \in \Gamma_N} \rho'(g) \mathbf{W}_{\alpha}(\tilde{\mathbf{f}}(x)) \partial^{\alpha}[\mathbf{f}](g^{-1}(x - t)) \quad (28)$$

Since the elementary PDOs are independent from each other and the arbitrariness of \mathbf{f} , the Eq.(26) and Eq.(28) equal if and only if their corresponding coefficient matrices of each $\partial^{\alpha}[\mathbf{f}](g^{-1}(x - t))$ is equal.

$$\forall \alpha \in \Gamma_N, \quad \sum_{\beta \in \Gamma_N} \hat{\rho}_{\beta, \alpha}(g) \mathbf{W}_{\beta}(\rho(g)\tilde{\mathbf{f}}(x)) \rho(g) = \rho'(g) \mathbf{W}_{\alpha}(\tilde{\mathbf{f}}(x)) \quad (29)$$

Due to the arbitrariness of $\mathbf{f}(x)$, the condition should be satisfied for any vector $\mathbf{y} \in \mathbb{R}^{c_{in}}$:

$$\forall \alpha \in \Gamma_N, \quad \sum_{\beta \in \Gamma_N} \hat{\rho}_{\beta, \alpha}(g) \mathbf{W}_{\beta}(\rho(g)\mathbf{y}) \rho(g) = \rho'(g) \mathbf{W}_{\alpha}(\mathbf{y}). \quad (30)$$

Hence, we get proposition.1.Q.E.D.

E PROOF OF PROPOSITION 2

Proposition 2 *Eq.(8) is equivalent to the following form:*

$$\forall g \in G, \forall \mathbf{y} \in \mathbb{R}^{c_{in}}, \quad \mathbf{W}(\rho(g)\mathbf{y})(\hat{\rho}(g) \otimes \rho(g)) = \rho'(g) \mathbf{W}(\mathbf{y}).$$

where \otimes is the tensor product, and $\forall \mathbf{y} \in \mathbb{R}^{c_{in}}, \mathbf{W}(\mathbf{y}) = [\mathbf{W}_{(0,0)}(\mathbf{y}), \dots, \mathbf{W}_{(0,N)}(\mathbf{y})]$.

Proof 3 Note that the dimension of coefficient matrices $\mathbf{W}_{\alpha}(\mathbf{y})$ is $c_{out} \times c_{in}$, and denote the i th coefficient matrix in the $\mathbf{W}(\mathbf{y})$ as $\mathbf{W}_{[i]}(\mathbf{y})$, such that $\mathbf{W}(\mathbf{y})_{m, ic_{in}+n} = (\mathbf{W}_{[i]}(\mathbf{y}))_{m,n}$.

According to proposition 1,

$$\begin{aligned}
& \forall 1 \leq i \leq |\Gamma_N|, \forall g \in G, \forall 1 \leq m \leq c_{out} \\
& \sum_{1 \leq j \leq |\Gamma_N|} \sum_{1 \leq n \leq c_{in}} \hat{\rho}_{ji}(g) (\mathbf{W}_{[j]}(\rho(g)\mathbf{y}))_{m,n} \rho_{nk}(g) = \sum_{1 \leq t \leq c_{in}} \rho'_{mt}(g) (\mathbf{W}_{[i]}(\mathbf{y}))_{t,k} \\
& \iff \sum_{1 \leq j \leq |\Gamma_N|} \sum_{1 \leq n \leq c_{in}} \mathbf{W}(\rho(g)\mathbf{y})_{m,jc_{in}+n} \hat{\rho}_{ji}(g) \rho_{nk}(g) = \sum_{1 \leq t \leq c_{in}} \rho'_{mt}(g) \mathbf{W}(\mathbf{y})_{t,ic_{in}+k} \\
& \iff \sum_{1 \leq j \leq |\Gamma_N|} \sum_{1 \leq n \leq c_{in}} \mathbf{W}(\rho(g)\mathbf{y})_{m,jc_{in}+n} (\hat{\rho}(g) \otimes \rho(g))_{jc_{in}+n,ic_{in}+k} = (\rho'(g) \mathbf{W}(\mathbf{y}))_{m,ic_{in}+k} \\
& \iff (\mathbf{W}(\rho(g)\mathbf{y}) (\hat{\rho}(g) \otimes \rho(g)))_{m,ic_{in}+k} = (\rho'(g) \mathbf{W}(\mathbf{y}))_{m,ic_{in}+k}.
\end{aligned} \tag{31}$$

Hence, we get the proposition 2. *Q.E.D.*

F PROOF OF PROPOSITION 3

Proposition 3 Suppose the input and output of operator (11) are both $\rho(g)$ -field. If the ρ is a regular or quotient representation of G , the constraint in Eq.(12) is equivalent to:

$$\forall g \in G, \forall \mathbf{y} \in \mathbb{R}^{c_{in}}, \quad \bar{\mathbf{w}}(\rho(g)\mathbf{y}) = (\rho(g) \otimes \hat{\rho}(g^{-1})^\top) \bar{\mathbf{w}}(\mathbf{y}),$$

where $\forall \mathbf{y} \in \mathbb{R}^{c_{in}}, \bar{\mathbf{w}}(\mathbf{y}) = \text{vec}([\mathbf{w}_{(0,0)}(\mathbf{y}), \dots, \mathbf{w}_{(0,N)}(\mathbf{y})])$ is a large vector concatenated from all generated vectors.

Before proving this proposition, we first prove the following lemma.

Lemma 2 For a n -dimensional permutation matrix \mathbf{P} and a n -dimensional vector \mathbf{w} , we have the following equation:

$$\mathbf{P} \text{diag}[\mathbf{w}] \mathbf{P}^{-1} = \text{diag}[\mathbf{P}\mathbf{w}], \tag{32}$$

where $\text{diag}[\cdot]$ is the operator that converting a n -dimensional vector to a n -dimensional diagonal matrix with the vector as diagonal.

Proof 4 Proving Eq.(32) is equivalent to proving the following:

$$\mathbf{P} \text{diag}[\mathbf{w}] = \text{diag}[\mathbf{P}\mathbf{w}] \mathbf{P} \tag{33}$$

Suppose the permutation matrix \mathbf{P} corresponds to a permutation: $\phi : \{1, \dots, n\} \rightarrow \{1, \dots, n\}$. Then we have $\mathbf{P}_{i,j} = \delta_{\phi(i),j}$. LHS of Eq.(33) is

$$(\mathbf{P} \text{diag}[\mathbf{w}])_{i,j} = \sum_{k=1}^n \delta_{\phi(i),k} w_k \delta_{k,j} = w_{\phi(i)} \delta_{\phi(i),j}. \tag{34}$$

The RHS of Eq.(33) is

$$(\text{diag}[\mathbf{P}\mathbf{w}] \mathbf{P})_{i,j} = \sum_{k=1}^n \sum_{l=1}^n \delta_{\phi(i),k} w_k \delta_{i,l} \delta_{\phi(l),j} \tag{35}$$

$$= \sum_{l=1}^n w_{\phi(i)} \delta_{i,l} \delta_{\phi(l),j} \tag{36}$$

$$= w_{\phi(i)} \delta_{\phi(i),j}. \tag{37}$$

So we have RHS=LHS which proves the lemma.

Proof 5 Applying the above lemma to the proposition 1, the Eq.(12) in the main text becomes:

$$\forall \alpha \in \Gamma_N, \forall g \in G, \forall \mathbf{y} \in \mathbb{R}^{c_{in}}, \quad \sum_{\beta \in \Gamma_N} \hat{\rho}_{\beta,\alpha}(g) \text{diag}[\mathbf{w}_\beta(\rho(g)\mathbf{y})] = \text{diag}[\rho(g) \mathbf{w}_\alpha(\mathbf{y})]. \tag{38}$$

We then prove this formula is equivalent to the following equation:

$$\forall g \in G, \forall \mathbf{y} \in \mathbb{R}^{c_{in}}, \quad \bar{\mathbf{W}}(\rho(g)\mathbf{y}) = \rho(g)\bar{\mathbf{W}}(\mathbf{y})\hat{\rho}(g^{-1}), \quad (39)$$

where $\bar{\mathbf{W}}$ is the matrix obtained by concating all the coefficient generator side by side such that $\forall \mathbf{y} \in \mathbb{R}^{c_{in}}, \bar{\mathbf{W}}(\mathbf{y}) = [\mathbf{w}_{(0,0)}(\mathbf{y}), \dots, \mathbf{w}_{(0,N)}(\mathbf{y})]$.

As proof 3, we denote the dimension of output from coefficient generator \mathbf{w}_α to be c_{in} -dimensional vector, and the i th coefficient generator concated in the $\bar{\mathbf{W}}$ as $\mathbf{w}_{[i]}$, such that $(\bar{\mathbf{W}}(\mathbf{y}))_{n,i} = (\mathbf{w}_{[i]}(\mathbf{y}))_n$.

Consider diagonals on both sides of Eq.(38), we have:

$$\begin{aligned} & \forall 1 \leq i \leq |\Gamma_N|, \forall g \in G, \forall 1 \leq m \leq c_{in}, \\ & \sum_{1 \leq j \leq |\Gamma_N|} \hat{\rho}_{ji}(g) \mathbf{w}_{[j]}(\rho(g)\mathbf{y}) = \rho(g) \mathbf{w}_{[i]}(\mathbf{y}) \\ \iff & \sum_{1 \leq j \leq |\Gamma_N|} \hat{\rho}_{ji}(g) (\mathbf{w}_{[j]}(\rho(g)\mathbf{y}))_m = \sum_{1 \leq t \leq c_{in}} \rho_{mt}(g) (\mathbf{w}_{[i]}(\mathbf{y}))_t \\ \iff & \sum_{1 \leq j \leq |\Gamma_N|} (\bar{\mathbf{W}}(\rho(g)\mathbf{y}))_{m,j} \hat{\rho}_{ji}(g) = \sum_{1 \leq t \leq c_{in}} \rho_{mt}(g) (\bar{\mathbf{W}}(\mathbf{y}))_{t,i} \\ \iff & (\bar{\mathbf{W}}(\rho(g)\mathbf{y})\hat{\rho}(g))_{m,i} = (\rho(g)\bar{\mathbf{W}}(\mathbf{y}))_{m,i} \\ \iff & \bar{\mathbf{W}}(\rho(g)\mathbf{y})\hat{\rho}(g) = \rho(g)\bar{\mathbf{W}}(\mathbf{y}) \\ \iff & \bar{\mathbf{W}}(\rho(g)\mathbf{y}) = \rho(g)\bar{\mathbf{W}}(\mathbf{y})\hat{\rho}(g^{-1}). \end{aligned} \quad (40)$$

Then, we get the Eq.(39). Applying $\text{vec}[\cdot]$ operator on both sides of Eq.(39), we get the proposition 3. *Q.E.D.*

G DETAILS ON MNIST-ROT EXPERIMENT

In all our models, we set $N = 4$ in Neural ePDOs for a trade-off between computation and accuracy. We show the architecture for C_{16} model of regular field in MNIST-rot experiment at Table 5.

Table 5: Architecture of Neural PDOs (C_{16}) on Mnist-rot classification, p means dropout rate.

Layer	Number of output fields
Steerable PDOs	16
BatchNorm+ReLU	
Neural ePDOs	24
BatchNorm+ReLU	
Spatial max pooling	
Neural ePDOs	32
BatchNorm+ReLU+Dropout($p=0.1$)	
Neural ePDOs	32
BatchNorm+ReLU+Dropout($p=0.1$)	
Spatial max pooling	
Neural ePDOs	48
BatchNorm+ReLU+Dropout($p=0.1$)	
Neural ePDOs	64
BatchNorm+ReLU+Dropout($p=0.1$)	
Group pooling	
Global max pooling	
Fully connected	64
BatchNorm+ReLU	
Fully connected+Softmax	10

We choose reduction ratio as $r = 1$, and partition number $q = \frac{z}{4}$ where z is number of input fields of current layer for the coefficient generator. For model of quotient representation, we adopt

$\rho_{reg}^{C_{16}} \oplus \rho_{quot}^{C_{16}/C_4}$ as the group representation and change the reduction ratio to $r = 2$. In addition, to keep the parameters almost invariant, we make a few modifications to the number of output fields at each layer. For the D_{16} regular representation model, we follow settings in (1) that keep the first five layers being D_{16} -equivariant, and restrict the final PDOs layer to be C_{16} -equivariant. The numbers of output fields of the six layers are 12, 16, 24, 24, 32, 128, while other hyperparameters are the same as C_{16} regular representation model. We choose $\sigma = 1$ for the Gaussian derivatives.

All our models are trained using the Adam optimizer (2) for 200 epochs with a batch size of 64. The learning rate is initialized as 0.002 and is reduced by 10 at the 60th, 120th and 160th epochs. The weight decay is set as 0.0001 and we augment the dataset with random rotations during training.

H DETAILS ON IMAGENET EXPERIMENT

For ImageNet100, we train the model using SGD optimizer for 80 epochs with momentum of 0.9, a batch size of 256 and weight decay of $1e-4$. The learning rate is initialized at 0.1 and divided by 5 after 20, 40, 60 epochs. Linear warm-up for 1 epoch with its ratio of 0.1 is adopted at the start of training. For each Neural PDOs, we set the reduction ratio $r = 2$, and the partition number $q = 10$ for the coefficient generator. We use random crops, horizontal flips as data augmentation. For ImageNet1k, we train the model for 100 epochs with learning rate initialized at 0.1 and divided by 10 at 30, 60, 90 epochs while keeping other setting invariant. We choose $\sigma = 1$ for the Gaussian derivatives.

I COMPARISON WITH METHOD IN STEERABLE PDOs

In steerable PDOs (1), they first make use of the duality between polynomials and PDOs to construct an isomorphic map from PDOs to matrices of polynomials, which transforms the problem of finding equivariant PDOs into the problem of solving polynomials based equivariant kernel. Then, they solve the polynomials based equivariant kernel by extending the solution basis of the equivariant kernel in E2CNN(3) to the polynomials form. However, such a method can not be easily extended to the non-linear PDOs proposed in the main paper, because there is no such isomorphic map to transform non-linear PDOs into matrices of polynomials. In comparison, we directly deduce the equivariant constraint on the coefficients of PDOs.

Note that the solution space of steerable PDOs is exactly included in the solution space of Eq.(9), as we choose the coefficient generator to be the constant function of the input.

In addition, our method no longer needs to compute the solution basis for each irreducible representation pairs manually beforehand, because the solving process is totally automatic for any representation pairs under our framework, which makes it much easier to extend to $G \leq O(n)$ for any $n \in \mathbb{N}$.

J HYPERPARAMETERS ANALYSIS

In this section, we investigate the influence of the hyperparameters change in Neural ePDOs. As pointed out in Section 6 in the main text, the number of our models' parameters is mainly controlled by the partition number p and the reduction ratio r , so we vary these two parameters respectively. The results are shown in Table 6 and Figure.2. We can see that all the Neural ePDOs models outperform steerable PDOs with significantly fewer parameters. Besides, as the size of Neural ePDOs is large enough, the performance gain is very limited, which demonstrates such a design can help to trade off efficiency and accuracy.

Table 6: Hyperparameters Analysis of Neural ePDOs. z is the number of input fields of the current layer. The test error with standard deviations are averaged over 5 runs.

hyperparameters	$p = \frac{z}{8}$	$p = \frac{z}{4}$	$p = \frac{z}{2}$
r=1	0.62±0.05	0.65± 0.04	0.67± 0.03
r=2	0.66±0.06	0.64±0.03	0.69± 0.03
r=4	0.66±0.04	0.67± 0.03	0.71± 0.04

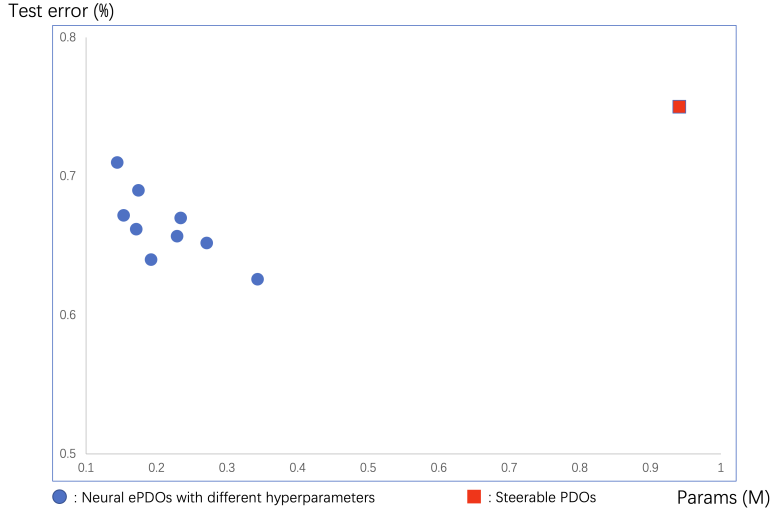


Figure 2: A comparison between Neural ePDOs of various model sizes and Steerable PDOs. The x-coordinate denotes the model parameters and the y-coordinate denotes the average test error. The blue dots are used to denote Neural ePDOs of various hyperparameters and the red square is used to denote Steerable PDOs.

K TRAINING AND TESTING TIME COMPARISON

In this section, we compare the training and testing time between Neural ePDOs and Steerable PDOs (1). We train and test models on a single GTX 1080 Ti. The training time is obtained by training model for 1 epoch, and the test time is the inference time for one image. The results are shown in Table 7.

Table 7: Training and testing time of Neural ePDOs compared to Steerable PDOs.

Model	Testing time(s)	Training time(s)
Steerable PDOs	0.0194	50.25
Neural ePDOs	0.0201	102.45

As shown in Table 7, the gap between the test time of Neural ePDOs and steerable PDOs is minor. However, the training time of steerable PDOs is shorter, despite Neural ePDOs having fewer FLOPs. It is because the steerable PDOs is implemented using convolution operators, which are highly optimized in existing speedup libraries. In comparison, we just give a naive implementation of Neural ePDOs without any support from these libraries. We leave implementing the specific CUDA kernel for GPU acceleration to further speed up the training process for Neural ePDOs as future work.

L FILTERS OF FINITE DIFFERENCE METHOD

The $\hat{\partial}$ is used to denote the discretization of PDOs.

L.1 FILTERS OF SIZE 3×3

$$\begin{aligned}
\dot{\partial}_0 &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} & \dot{\partial}_x &= \frac{1}{h} \begin{bmatrix} 0 & 0 & 0 \\ -\frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 \end{bmatrix} & \dot{\partial}_y &= \frac{1}{h} \begin{bmatrix} 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 \\ 0 & -\frac{1}{2} & 0 \end{bmatrix} \\
\dot{\partial}_{xx} &= \frac{1}{h^2} \begin{bmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{bmatrix} & \dot{\partial}_{xy} &= \frac{1}{h^2} \begin{bmatrix} -\frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 \\ \frac{1}{4} & 0 & -\frac{1}{4} \end{bmatrix} & \dot{\partial}_{yy} &= \frac{1}{h^2} \begin{bmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{bmatrix} \\
\dot{\partial}_{xxy} &= \frac{1}{h^3} \begin{bmatrix} \frac{1}{2} & -1 & \frac{1}{2} \\ 0 & 0 & 0 \\ -\frac{1}{2} & 1 & -\frac{1}{2} \end{bmatrix} & \dot{\partial}_{xyy} &= \frac{1}{h^3} \begin{bmatrix} -\frac{1}{2} & 0 & \frac{1}{2} \\ 1 & 0 & -1 \\ -\frac{1}{2} & 0 & \frac{1}{2} \end{bmatrix} & \dot{\partial}_{xyy} &= \frac{1}{h^4} \begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}
\end{aligned}$$

L.2 FILTERS OF SIZE 5×5

$$\begin{aligned}
\dot{\partial}_{xxx} &= \frac{1}{h^3} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -\frac{1}{2} & 1 & 0 & -1 & \frac{1}{2} \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} & \dot{\partial}_{yyy} &= \frac{1}{h^3} \begin{bmatrix} 0 & 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{1}{2} & 0 & 0 \end{bmatrix} \\
\dot{\partial}_{xxx} &= \frac{1}{h^4} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 6 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} & \dot{\partial}_{xxy} &= \frac{1}{h^4} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ -\frac{1}{4} & \frac{1}{2} & 0 & -\frac{1}{2} & \frac{1}{4} \\ 0 & 0 & 0 & 0 & 0 \\ \frac{1}{4} & -\frac{1}{2} & 0 & \frac{1}{2} & -\frac{1}{4} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \\
\dot{\partial}_{yyy} &= \frac{1}{h^4} \begin{bmatrix} 0 & -\frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & \frac{1}{2} & 0 & -\frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{4} & 0 & -\frac{1}{4} & 0 \end{bmatrix} & \dot{\partial}_{yyy} &= \frac{1}{h^4} \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -4 & 0 & 0 \\ 0 & 0 & 6 & 0 & 0 \\ 0 & 0 & -4 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}
\end{aligned}$$

REFERENCES

- [1] Erik Jenner and Maurice Weiler. Steerable partial differential operators for equivariant neural networks. *arXiv preprint arXiv:2106.10163*, 2021.
- [2] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [3] Maurice Weiler and Gabriele Cesa. General E (2)-equivariant steerable CNNs. *Advances in Neural Information Processing Systems*, 32, 2019.