Supplement to "Monomial Matrix Group Equivariant Neural Functional Networks"

Table of Contents

A Construction of Monomial Matrix Group Equivariant Layers

In this appendix, we present how we constructed Monomial Matrix Group Equivariant Layers. We adopt the idea of notation in [71] to derive the formula of linear functional layers. For two weight spaces $\mathcal{U}$ and $\mathcal{U}'$ with the same number of layers L as well as the same number of channels at i-th layer n_i:

$$\begin{aligned} \mathcal{U} &= \mathcal{W} \times \mathcal{B} \text{ where:} \\ \mathcal{W} &= \mathbb{R}^{w_L \times n_L \times n_{L-1}} \times \ldots \times \mathbb{R}^{w_2 \times n_2 \times n_1} \times \mathbb{R}^{w_1 \times n_1 \times n_0}, \\ \mathcal{B} &= \mathbb{R}^{b_L \times n_L \times 1} \times \ldots \times \mathbb{R}^{b_2 \times n_2 \times 1} \times \mathbb{R}^{b_1 \times n_1 \times 1}; \end{aligned} \tag{22}$$

and

$$\begin{aligned} \mathcal{U}' &= \mathcal{W}' \times \mathcal{B}' \text{ where:} \\ \mathcal{W}' &= \mathbb{R}^{w'_L \times n_L \times n_{L-1}} \times \ldots \times \mathbb{R}^{w'_2 \times n_2 \times n_1} \times \mathbb{R}^{w'_1 \times n_1 \times n_0}, \\ \mathcal{B}' &= \mathbb{R}^{b'_L \times n_L \times 1} \times \ldots \times \mathbb{R}^{b'_2 \times n_2 \times 1} \times \mathbb{R}^{b'_1 \times n_1 \times 1}; \end{aligned} \tag{23}$$

our equivariant layer $E \colon \mathcal{U} \to \mathcal{U}'$ will has the form as follows:

$$E \quad : (W, b) = U \longmapsto U' = (W', b') \text{ where:} \tag{24}$$

$$W'^{(i)}_{jk} := \sum_{s=1}^{L} \sum_{p=1}^{n_s} \sum_{q=1}^{n_{s-1}} \mathfrak{p}^{ijk}_{spq} W^{(s)}_{pq} + \sum_{s=1}^{L} \sum_{p=1}^{n_s} \mathfrak{q}^{ijk}_{sp} b^{(s)}_{p} + \mathfrak{t}^{ijk} \tag{25}$$

$$b'^{(i)}_{j} := \sum_{s=1}^{L} \sum_{p=1}^{n_s} \sum_{q=1}^{n_{s-1}} \mathfrak{r}^{ij}_{spq} W^{(s)}_{pq} + \sum_{s=1}^{L} \sum_{p=1}^{n_s} \mathfrak{s}^{ij}_{sp} b^{(s)}_{p} + \mathfrak{t}^{ij} \tag{26}$$

Here, the map E is parameterized by hyperparameter $\theta = (\mathfrak{p}, \mathfrak{q}, \mathfrak{s}, \mathfrak{r}, \mathfrak{t})$ with dimensions of each component as follows:

- $\mathfrak{p}^{ijk}_{spq} \in \mathbb{R}^{w'_i \times w_s}$ represents the contribution of $W^{(s)}_{pq}$ to $W'^{(i)}_{jk}$,

- $\mathfrak{q}^{ijk}_{sp} \in \mathbb{R}^{w'_i \times b_s}$ represents the contribution of $b^{(s)}_p$ to $W'^{(i)}_{jk}$,
- $\mathfrak{t}^{ijk} \in \mathbb{R}^{w'_i}$ is the bias of the layer for $W'^{(i)}_{jk}$;
- $\mathfrak{r}^{ij}_{spq} \in \mathbb{R}^{b'_i \times w_s}$ represents the contribution of $W^{(s)}_{pq}$ to $b'^{(i)}_j$,
- $\mathfrak{s}^{ij}_{sp} \in \mathbb{R}^{b'_i \times b_s}$ represents the contribution of $b^{(s)}_p$ to $b'^{(i)}_j$,
- $\mathfrak{t}^{ij} \in \mathbb{R}^{b'_i}$ is the bias of the layer for $b'^{(i)}_j$.

We want to see how an element of the group $\mathcal{G}_{\mathcal{U}}$ acts on input and output of layer E. Let

$$g = \left(g^{(L)}, \ldots, g^{(0)}\right) \in \mathcal{G}_{n_L} \times \ldots \times \mathcal{G}_{n_0} = \mathcal{G}_{\mathcal{U}}, \tag{27}$$

where

$$g^{(i)} = D^{(i)} \cdot P_{\pi_i} = \operatorname{diag}\left(d^{(i)}_1, \ldots, d^{(i)}_{n_i}\right) \cdot P_{\pi_i} \in \mathcal{G}_{n_i}. \tag{28}$$

Recall the definition of the group action $gU = (gW, gb)$ where:

$$(gW)^{(i)} := \left(g^{(i)}\right) \cdot W^{(i)} \cdot \left(g^{(i-1)}\right)^{-1} \quad \text{and} \quad (gb)^{(i)} := \left(g^{(i)}\right) \cdot b^{(i)}, \tag{29}$$

or in term of entries:

$$(gW)^{(i)}_{jk} := \frac{d^{(i)}_j}{d^{(i-1)}_k} \cdot W^{(i)}_{\pi_i^{-1}(j)\pi_{i-1}^{-1}(k)} \quad \text{and} \quad (gb)^{(i)}_j := d^{(i)}_j \cdot b^{(i)}_{\pi_i^{-1}(j)}. \tag{30}$$

$gE(U) = gU' = (gW', gb')$ is computed as follows:

$$\begin{aligned}
(gW')^{(i)}_{jk} &= \frac{d^{(i)}_j}{d^{(i-1)}_k} \cdot W'^{(i)}_{\pi_i^{-1}(j)\pi_{i-1}^{-1}(k)} && (31)\\
&= \frac{d^{(i)}_j}{d^{(i-1)}_k} \cdot \left(\sum_{s=1}^{L} \sum_{p=1}^{n_s} \sum_{q=1}^{n_{s-1}} \mathfrak{p}^{i\pi_i^{-1}(j)\pi_{i-1}^{-1}(k)}_{spq} W^{(s)}_{pq} + \right. && (32)\\
&\qquad \left. \sum_{s=1}^{L} \sum_{p=1}^{n_s} \mathfrak{q}^{i\pi_i^{-1}(j)\pi_{i-1}^{-1}(k)}_{sp} b^{(s)}_p + \mathfrak{t}^{i\pi_i^{-1}(j)\pi_{i-1}^{-1}(k)} \right) && (33)\\
(gb')^{(i)}_j &= d^{(i)}_j \cdot b'^{(i)}_{\pi_i^{-1}(j)} && (34)\\
&= d^{(i)}_j \cdot \left(\sum_{s=1}^{L} \sum_{p=1}^{n_s} \sum_{q=1}^{n_{s-1}} \mathfrak{s}^{i\pi_i^{-1}(j)}_{spq} W^{(s)}_{pq} + \right. && (35)\\
&\qquad \left. \sum_{s=1}^{L} \sum_{p=1}^{n_s} \mathfrak{r}^{i\pi_i^{-1}(j)}_{sp} b^{(s)}_p + \mathfrak{t}^{i\pi_i^{-1}(j)} \right). && (36)
\end{aligned}$$

$E(gU) = (gU)' = ((gW)', (gU)')$ is computed as follows:

$$(gU)'^{(i)}_{jk} = \sum_{s=1}^{L}\sum_{p=1}^{n_s}\sum_{q=1}^{n_{s-1}} \mathfrak{p}^{ijk}_{spq} \cdot \frac{d^{(s)}_p}{d^{(s-1)}_q} \cdot W^{(s)}_{\pi_s^{-1}(p)\pi_{s-1}^{-1}(q)} + \sum_{s=1}^{L}\sum_{p=1}^{n_s} \mathfrak{q}^{ijk}_{sp} \cdot d^{(s)}_p \cdot b^{(s)}_{\pi_s^{-1}(p)} + \mathfrak{t}^{ijk} \tag{37}$$

$$= \sum_{s=1}^{L}\sum_{p=1}^{n_s}\sum_{q=1}^{n_{s-1}} \mathfrak{p}^{ijk}_{s\pi_s(p)\pi_{s-1}(q)} \cdot \frac{d^{(s)}_{\pi_s(p)}}{d^{(s-1)}_{\pi_{s-1}(q)}} \cdot W^{(s)}_{pq} + \sum_{s=1}^{L}\sum_{p=1}^{n_s} \mathfrak{q}^{ijk}_{s\pi_s(p)} \cdot d^{(s)}_{\pi_s(p)} \cdot b^{(s)}_p + \mathfrak{t}^{ijk} \tag{38}$$

$$(gb)'^{(i)}_{j} = \sum_{s=1}^{L}\sum_{p=1}^{n_s}\sum_{q=1}^{n_{s-1}} \mathfrak{r}^{ij}_{spq} \cdot \frac{d^{(s)}_p}{d^{(s-1)}_q} \cdot W^{(s)}_{\pi_s^{-1}(p)\pi_{s-1}^{-1}(q)} + \sum_{s=1}^{L}\sum_{p=1}^{n_s} \mathfrak{s}^{ij}_{sp} \cdot d^{(s)}_p \cdot b^{(s)}_{\pi_s^{-1}(p)} + \mathfrak{t}^{ij} \tag{39}$$

$$= \sum_{s=1}^{L}\sum_{p=1}^{n_s}\sum_{q=1}^{n_{s-1}} \mathfrak{r}^{ij}_{s\pi_s(p)\pi_{s-1}(q)} \cdot \frac{d^{(s)}_{\pi_s(p)}}{d^{(s-1)}_{\pi_{s-1}(q)}} \cdot W^{(s)}_{pq} + \sum_{s=1}^{L}\sum_{p=1}^{n_s} \mathfrak{s}^{ij}_{s\pi_s(p)} \cdot d^{(s)}_{\pi_s(p)} \cdot b^{(s)}_p + \mathfrak{t}^{ij}. \tag{40}$$

We need E is G-equivariant under the action of subgroups of $\mathcal{G}_{\mathcal{U}}$ as in Theorem 4.4. From the above computation, if $gE(U) = E(gU)$, the hyperparameter $\theta = (\mathfrak{p}, \mathfrak{q}, \mathfrak{r}, \mathfrak{s}, \mathfrak{t})$ have to satisfy the system of constraints as follows:

$$\frac{d^{(i)}_j}{d^{(i-1)}_k} \cdot \mathfrak{p}^{i\pi_i^{-1}(j)\pi_{i-1}^{-1}(k)}_{spq} = \mathfrak{p}^{ijk}_{s\pi_s(p)\pi_{s-1}(q)} \cdot \frac{d^{(s)}_{\pi_s(p)}}{d^{(s-1)}_{\pi_{s-1}(q)}} \tag{41}$$

$$\frac{d^{(i)}_j}{d^{(i-1)}_k} \cdot \mathfrak{q}^{i\pi_i^{-1}(j)\pi_{i-1}^{-1}(k)}_{sp} = \mathfrak{q}_{s\pi_s(p)} \cdot d^{(s)}_{\pi_s(p)} \tag{42}$$

$$d^{(i)}_j \cdot \mathfrak{r}^{i\pi_i^{-1}(j)}_{spq} = \mathfrak{r}^{ij}_{s\pi_s(p)\pi_{s-1}(q)} \cdot \frac{d^{(s)}_{\pi_s(p)}}{d^{(s-1)}_{\pi_{s-1}(q)}} \tag{43}$$

$$d^{(i)}_j \cdot \mathfrak{s}^{i\pi_i^{-1}(j)}_{sp} = \mathfrak{s}^{ij}_{s\pi_s(p)} \cdot d^{(s)}_{\pi_s(p)} \tag{44}$$

$$\frac{d^{(i)}_j}{d^{(i-1)}_k} \cdot \mathfrak{t}^{i\pi_i^{-1}(j)\pi_{i-1}^{-1}(k)} = \mathfrak{t}^{ijk} \tag{45}$$

$$d^{(i)}_j \cdot \mathfrak{t}^{i\pi_i^{-1}(j)} = \mathfrak{t}^{ij}. \tag{46}$$

for all possible tuples $((i,j,k),(s,p,q))$ and all $g \in G$. Since the two subgroups G considered in Theorem 4.4 satisfy that: $G \cap \mathcal{P}_i$ is trivial (for $i = 0$ or $i = L$) or the whole $\mathcal{P}_i$ (for $0 < i < L$), so we can simplify the above system of constraints by moving all the permutation π's to LHS, then replacing π^{-1} by π. The system, denoted as (*), now is written as follows:

$$\frac{d^{(i)}_j}{d^{(i-1)}_k} \cdot \mathfrak{p}^{i\pi_i(j)\pi_{i-1}(k)}_{s\pi_s(p)\pi_{s-1}(q)} = \mathfrak{p}^{ijk}_{spq} \cdot \frac{d^{(s)}_p}{d^{(s-1)}_q} \tag{*1}$$

$$\frac{d^{(i)}_j}{d^{(i-1)}_k} \cdot \mathfrak{q}^{i\pi_i(j)\pi_{i-1}(k)}_{s\pi_s(p)} = \mathfrak{q}^{ijk}_{sp} \cdot d^{(s)}_p \tag{*2}$$

$$d^{(i)}_j \cdot \mathfrak{r}^{i\pi_i(j)}_{s\pi_s(p)\pi_{s-1}(q)} = \mathfrak{r}^{ij}_{spq} \cdot \frac{d^{(s)}_p}{d^{(s-1)}_q} \tag{*3}$$

$$d^{(i)}_j \cdot \mathfrak{s}^{i\pi_i(j)}_{s\pi_s(p)} = \mathfrak{s}^{ij}_{sp} \cdot d^{(s)}_p \tag{*4}$$

$$\frac{d^{(i)}_j}{d^{(i-1)}_k} \cdot \mathfrak{t}^{i\pi_i^{-1}(j)\pi_{i-1}^{-1}(k)} = \mathfrak{t}^{ijk} \tag{*5}$$

$$d^{(i)}_j \cdot \mathfrak{t}^{i\pi_i^{-1}(j)} = \mathfrak{t}^{ij} \tag{*6}$$

We treat each case of activation separately.

Table 5: Hyperparameter of Equivariant Layers with ReLU activation. *Left* presents all possible case of tuple $((i, j, k), (s, p, q))$, and *Right* presents the parameter at the corresponding position. Here, we have three types of notations: 0 means the parameter equal to 0; equations with π's in LHS means the equation holds for all possible π; and a single term with no further information means the term can be arbitrary.

Tuple $((i,j,k),(s,p,q))$			Hyperparameter $(\mathfrak{p},\mathfrak{q},\mathfrak{r},\mathfrak{s})$			
i and s	j and p	k and q	$\mathfrak{p}^{ijk}_{spq}$	$\mathfrak{q}^{ijk}_{sp}$	$\mathfrak{p}^{ij}_{spq}$	$\mathfrak{p}^{ij}_{sp}$
$i = s = 1$	$j \neq p$		0	0	0	0
	$j = p$		$\mathfrak{p}^{1\pi(j)k}_{1\pi(j)q} = \mathfrak{p}^{1jk}_{1jq}$	$\mathfrak{q}^{1\pi(j)k}_{1\pi(j)} = \mathfrak{q}^{1jk}_{1j}$	$\mathfrak{r}^{1\pi(j)}_{1\pi(j)q} = \mathfrak{r}^{1j}_{1jq}$	$\mathfrak{s}^{1\pi(j)}_{1\pi(j)} = \mathfrak{s}^{1j}_{1j}$
$i = s = L$		$k \neq q$	0	0	0	$\mathfrak{s}^{Lj}_{Lp}$
		$k = q$	$\mathfrak{p}^{Lj\pi(k)}_{Lp\pi(k)} = \mathfrak{p}^{Ljk}_{Ljq}$	0	0	$\mathfrak{s}^{Lj}_{Lp}$
$1 < i = s < L$	$j \neq p$		0	0	0	0
	$j = p$	$k \neq q$	0	0	0	$\mathfrak{s}^{i\pi(j)}_{i\pi(j)} = \mathfrak{s}^{ij}_{ij}$
		$k = q$	$\mathfrak{p}^{i\pi(j)\pi'(k)}_{i\pi(j)\pi'(k)} = \mathfrak{p}^{ijk}_{ijk}$	0	0	$\mathfrak{s}^{i\pi(j)}_{i\pi(j)} = \mathfrak{s}^{ij}_{ij}$
$i \neq s$			0	0	0	0

Table 6: Construction of equivariant functional layer with ReLU activation. Note that all parameters have to satisfy the conditions presented in Table 5.

Layer	Equivariant layer $E : (W, b) \longmapsto (W', b')$	
	$W'^{(i)}_{jk}$	$b'^{(i)}_{j}$
$i = 1$	$\sum_{q=1}^{n_0} \mathfrak{p}^{1jk}_{1jq} W^{(1)}_{jq} + \mathfrak{q}^{1jk}_{1j} b^{(1)}_{j}$	$\sum_{q=1}^{n_0} \mathfrak{r}^{1j}_{1jq} W^{(1)}_{jq} + \mathfrak{s}^{1j}_{1j} b^{(1)}_{j}$
$1 < i < L$	$\mathfrak{p}^{ijk}_{ijk} W^{(i)}_{jk}$	$\mathfrak{s}^{ij}_{ij} b^{(i)}_{j}$
$i = L$	$\sum_{p=1}^{n_L} \mathfrak{p}^{Ljk}_{Lpk} W^{(L)}_{pk}$	$\sum_{p=1}^{n_L} \mathfrak{s}^{Lj}_{Lp} b^{(L)}_{p} + \mathfrak{t}^{Lj}$

A.1 ReLU activation

Recall that, in this case:

$$G \coloneqq \{\mathrm{id}_{\mathcal{G}_{n_L}}\} \times \mathcal{G}^{>0}_{n_{L-1}} \times \ldots \times \mathcal{G}^{>0}_{n_1} \times \{\mathrm{id}_{\mathcal{G}_{n_0}}\}. \tag{47}$$

So the system of constraints (*) holds for:

1. all possible tuples $((i, j, k), (s, p, q))$,
2. all $\pi_i \in \mathcal{P}_i$ for $0 < i < L$, all $d^{(i)}_j > 0$ for $0 < i < L$, $1 \leqslant j \leqslant n_i$,
3. $\pi_i = \mathrm{id}_{\mathcal{G}_{n_i}}$ and $d^{(i)}_j = 1$ for $i = 0$ or $i = L$.

By treat each case of tuples $((i, j, k), (s, p, q))$, we solve Eq. *1, Eq. *2, Eq. *3, Eq. *4 in the system (*) for hyperparameter $(\mathfrak{p}, \mathfrak{q}, \mathfrak{r}, \mathfrak{s})$ as in Table 5. For $\mathfrak{t}^{ijk}$ and $\mathfrak{t}^{ij}$, by Eq. *5, Eq. *6, we have $\mathfrak{t}^{ijk} = 0$ for all (i, j, k), $\mathfrak{t}^{ij} = 0$ if $i < L$, and $\mathfrak{t}^{Lj}$ is arbitrary for all $1 \leqslant j \leqslant n_L$. In conclusion, the formula of equivariant layers E in case of activation ReLU is presented as in Table 6.

Example A.1. Let us consider a two-hidden-layers MLP with activation $\sigma = \mathrm{ReLU}$. Assume that $n_0 = n_1 = n_2 = n_3 = 2$, i.e., all layers have two neurons. This MLP defines a function $f : \mathbb{R}^2 \to \mathbb{R}^2$ given by

$$f(x) = W^{(3)}\sigma\left(W^{(2)}\sigma\left(W^{(1)}x + b^{(1)}\right) + b^{(2)}\right) + b^{(3)},$$

where $W^{(i)} = \begin{pmatrix} W^{(i)}_{11} & W^{(i)}_{12} \\ W^{(i)}_{21} & W^{(i)}_{22} \end{pmatrix}$ is a 2×2 matrix and $b^{(i)} = \begin{bmatrix} b^{(i)}_1 \\ b^{(i)}_2 \end{bmatrix}$ for each $i = 1, 2, 3$. In this case, the weight space $\mathcal{U}$ consists of the tuples

$$U = (W^{(1)}, W^{(2)}, W^{(3)}, b^{(1)}, b^{(2)}, b^{(3)})$$

and it has dimension 18.

According to Eq. (27), an equivariant layer E over $\mathcal{U}$ has the form

$$E(U) = \left(W'^{(1)}, W'^{(2)}, W'^{(3)}, b'^{(1)}, b'^{(2)}, b'^{(3)}\right),$$

where

$$\begin{aligned}
W'^{(1)}_{jk} &= \mathfrak{p}^{1jk}_{1j_1} W^{(1)}_{j_1 1} + \mathfrak{p}^{1jk}_{1j_2} W^{(1)}_{j_2 2} + \mathfrak{q}^{1jk}_{1j} b^{(1)}_j, & b'^{(1)}_j &= \mathfrak{r}^{1j}_{j_1} W^{(1)}_{j_1 1} + \mathfrak{r}^{1j}_{j_2} W^{(1)}_{j_2 2} + \mathfrak{s}^{1j}_{1j} b^{(1)}_j, \\
W'^{(2)}_{jk} &= \mathfrak{p}^{2jk}_{2j} W^{(2)}_{jk}, & b'^{(2)}_j &= \mathfrak{s}^{2j}_{2j} b^{(2)}_j, \\
W'^{(3)}_{jk} &= \mathfrak{p}^{3jk}_{3k_1} W^{(3)}_{3k} + \mathfrak{p}^{3jk}_{3k_2} W^{(3)}_{2k}, & b'^{(3)}_j &= \mathfrak{s}^{3j}_{3j_1} b^{(3)}_1 + \mathfrak{s}^{3j}_{3j_2} b^{(3)}_2 + \mathfrak{r}^3_j.
\end{aligned}$$

These equations can be written in a friendly matrix form as follows.

$$\begin{bmatrix} W'^{(1)}_{11} \\ W'^{(1)}_{12} \\ W'^{(1)}_{21} \\ W'^{(1)}_{22} \\ b'^{(1)}_1 \\ b'^{(1)}_2 \end{bmatrix} = \begin{bmatrix} \mathfrak{p}^{111}_{111} & \mathfrak{p}^{111}_{112} & 0 & 0 & \mathfrak{q}^{111}_{111} & 0 \\ \mathfrak{p}^{112}_{111} & \mathfrak{p}^{112}_{112} & 0 & 0 & \mathfrak{q}^{112}_{111} & 0 \\ 0 & 0 & \mathfrak{p}^{121}_{121} & \mathfrak{p}^{121}_{122} & 0 & \mathfrak{q}^{121}_{112} \\ 0 & 0 & \mathfrak{p}^{122}_{121} & \mathfrak{p}^{122}_{122} & 0 & \mathfrak{q}^{122}_{112} \\ \mathfrak{r}^{111}_{111} & \mathfrak{r}^{111}_{112} & 0 & 0 & \mathfrak{s}^{111}_{111} & 0 \\ 0 & 0 & \mathfrak{r}^{121}_{121} & \mathfrak{r}^{122}_{122} & 0 & \mathfrak{s}^{112}_{112} \end{bmatrix} \begin{bmatrix} W^{(1)}_{11} \\ W^{(1)}_{12} \\ W^{(1)}_{21} \\ W^{(1)}_{22} \\ b^{(1)}_1 \\ b^{(1)}_2 \end{bmatrix},$$

$$\begin{bmatrix} W'^{(2)}_{11} \\ W'^{(2)}_{12} \\ W'^{(2)}_{21} \\ W'^{(2)}_{22} \\ b'^{(2)}_1 \\ b'^{(2)}_2 \end{bmatrix} = \begin{bmatrix} \mathfrak{p}^{211}_{211} & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathfrak{p}^{212}_{212} & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathfrak{p}^{221}_{221} & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathfrak{p}^{222}_{222} & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathfrak{s}^{211}_{211} & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathfrak{s}^{222}_{222} \end{bmatrix} \begin{bmatrix} W^{(2)}_{11} \\ W^{(2)}_{12} \\ W^{(2)}_{21} \\ W^{(2)}_{22} \\ b^{(2)}_1 \\ b^{(2)}_2 \end{bmatrix},$$

$$\begin{bmatrix} W'^{(3)}_{11} \\ W'^{(3)}_{12} \\ W'^{(3)}_{21} \\ W'^{(3)}_{22} \\ b'^{(3)}_1 \\ b'^{(3)}_2 \end{bmatrix} = \begin{bmatrix} \mathfrak{p}^{311}_{311} & 0 & \mathfrak{p}^{311}_{321} & 0 & 0 & 0 \\ 0 & \mathfrak{p}^{312}_{312} & 0 & \mathfrak{p}^{322}_{322} & 0 & 0 \\ \mathfrak{p}^{321}_{312} & 0 & \mathfrak{p}^{321}_{321} & 0 & 0 & 0 \\ 0 & \mathfrak{p}^{322}_{312} & 0 & \mathfrak{p}^{322}_{322} & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathfrak{s}^{311}_{311} & \mathfrak{s}^{312}_{312} \\ 0 & 0 & 0 & 0 & \mathfrak{s}^{321}_{321} & \mathfrak{s}^{322}_{322} \end{bmatrix} \begin{bmatrix} W^{(3)}_{11} \\ W^{(3)}_{12} \\ W^{(3)}_{21} \\ W^{(3)}_{22} \\ b^{(3)}_1 \\ b^{(3)}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \mathfrak{r}^3_1 \\ \mathfrak{r}^3_2 \end{bmatrix}.$$

A.2 Sin or Tanh activation

Recall that, in this case:

$$G \coloneqq \{\mathrm{id}_{\mathcal{G}_{n_L}}\} \times \mathcal{G}^{\pm 1}_{n_{L-1}} \times \ldots \times \mathcal{G}^{\pm 1}_{n_1} \times \{\mathrm{id}_{\mathcal{G}_{n_0}}\}. \tag{48}$$

So the system of constraints (*) holds for:

1. all possible tuples $((i, j, k), (s, p, q))$,
2. all $\pi_i \in \mathcal{P}_i$ for $0 < i < L$, all $d^{(i)}_j \in \{\pm 1\}$ for $0 < i < L$, $1 \leqslant j \leqslant n_i$,
3. $\pi_i = \mathrm{id}_{\mathcal{G}_{n_i}}$ and $d^{(i)}_j = 1$ for $i = 0$ or $i = L$.

We assume $L \geqslant 3$, the case $L \leqslant 2$ can be solved similarly. By treat each case of tuples $((i, j, k), (s, p, q))$, we solve Eq. *1, Eq. *2, Eq. *3, Eq. *4 in the system (*) for hyperparameter $(\mathfrak{p}, \mathfrak{q}, \mathfrak{r}, \mathfrak{s})$ as in Table 7. For $\mathfrak{t}^{ijk}$ and $\mathfrak{t}^{ij}$, by Eq. *5, Eq. *6, we have $\mathfrak{t}^{ijk} = 0$ for all (i, j, k), $\mathfrak{t}^{ij} = 0$ if $i < L$, and $\mathfrak{t}^{Lj}$ is arbitrary for all $1 \leqslant j \leqslant n_L$. In conclusion, the formula of equivariant layers E in case of $\sin$ or Tanh activation is presented as in Table 8.

Table 7: Hyperparameter of Equivariant Layers with $\sin$ or Tanh activation. *Left* presents all possible case of tuple $((i,j,k),(s,p,q))$, and *Right* presents the parameter at the corresponding position. Here, we have three types of notations: 0 means the parameter equal to 0; equations with π's in LHS means the equation holds for all possible π; and a single term with no further information means the term can be arbitrary.

Tuple $((i,j,k),(s,p,q))$			Hyperparameter $(\mathfrak{p},\mathfrak{q},\mathfrak{r},\mathfrak{s})$			
i and s	j and p	k and q	$\mathfrak{p}^{ijk}_{spq}$	$\mathfrak{q}^{ijk}_{sp}$	$\mathfrak{r}^{ij}_{spq}$	$\mathfrak{s}^{ij}_{sp}$
$i=s=1$	$j\neq p$		0	0	0	0
	$j=p$		$\mathfrak{p}^{1\pi(j)k}_{1\pi(j)q}=\mathfrak{p}^{1jk}_{1jq}$	$\mathfrak{q}^{1\pi(j)k}_{1\pi(j)}=\mathfrak{q}^{1jk}_{1j}$	$\mathfrak{r}^{1\pi(j)}_{1\pi(j)q}=\mathfrak{r}^{1j}_{1jq}$	$\mathfrak{s}^{1\pi(j)}_{1\pi(j)}=\mathfrak{s}^{1j}_{1j}$
$i=s=L$		$k\neq q$	0	0	0	$\mathfrak{s}^{Lj}_{Lp}$
		$k=q$	$\mathfrak{p}^{Lj\pi(k)}_{Lp\pi(k)}=\mathfrak{p}^{Ljk}_{Ljq}$	0	0	$\mathfrak{s}^{Lj}_{Lp}$
$1<i=s<L$	$j\neq p$		0	0	0	0
	$j=p$	$k\neq q$	0	0	0	$\mathfrak{s}^{i\pi(j)}_{i\pi(j)}=\mathfrak{s}^{ij}_{ij}$
		$k=q$	$\mathfrak{p}^{i\pi(j)\pi'(k)}_{i\pi(j)\pi'(k)}=\mathfrak{p}^{ijk}_{ijk}$	0	0	$\mathfrak{s}^{i\pi(j)}_{i\pi(j)}=\mathfrak{s}^{ij}_{ij}$
$(i,s)=(L-1,L)$	$j=q$		0	0	$\mathfrak{r}^{(L-1)\pi(j)}_{Lp\pi(j)}=\mathfrak{r}^{(L-1)j}_{Lpj}$	0
$(i,s)=(L,L-1)$	$k=p$		0	$\mathfrak{q}^{Lj\pi(k)}_{(L-1)\pi(k)}=\mathfrak{q}^{Ljk}_{(L-1)k}$	0	0
otherwise			0	0	0	0

Table 8: Construction of equivariant functional layer with $\sin$ or Tanh activation. Note that all parameters have to satisfy the conditions presented in Table 5.

Layer	Equivariant layer $E\,:\,(W,b)\longmapsto(W',b')$	
	$W'^{(i)}_{jk}$	$b'^{(i)}_{j}$
$i=1$	$\sum_{q=1}^{n_0}\mathfrak{p}^{1jk}_{1jq}W^{(1)}_{jq}+\mathfrak{q}^{1jk}_{1j}b^{(1)}_{j}$	$\sum_{q=1}^{n_0}\mathfrak{r}^{1j}_{1jq}W^{(1)}_{jq}+\mathfrak{s}^{1j}_{1j}b^{(1)}_{j}$
$1<i<L-1$	$\mathfrak{p}^{ijk}_{ijk}W^{(i)}_{jk}$	$\mathfrak{s}^{ij}_{ij}b^{(i)}_{j}$
$i=L-1$	$\mathfrak{p}^{(L-1)jk}_{(L-1)jk}W^{(L-1)}_{jk}$	$\sum_{p=1}^{n_L}\mathfrak{r}^{(L-1)j}_{Lpj}W^{(L)}_{pj}+\mathfrak{s}^{(L-1)j}_{(L-1)j}b^{(L-1)}_{j}$
$i=L$	$\sum_{p=1}^{n_L}\mathfrak{p}^{Ljk}_{Lpk}W^{(L)}_{pk}+\mathfrak{q}^{Ljk}_{(L-1)k}b^{(L-1)}_{k}$	$\sum_{p=1}^{n_L}\mathfrak{s}^{Lj}_{Lp}b^{(L)}_{p}+\mathfrak{t}^{Lj}$

B Construction of Monomial Matrix Group Invariant Layers

In this appendix, we present how we constructed Monomial Matrix Group Invariant Layers. Let $\mathcal{U}$ be a weight spaces with the number of layers L as well as the number of channels at i-th layer n_i. We want to construct G-invariant layers $I\,:\,\mathcal{U}\to\mathbb{R}^d$ for some $d>0$. We treat each case of activations separately.

B.1 ReLU activation

Recall that, in this case:

$$G\coloneqq\{\mathrm{id}_{\mathcal{G}_{n_L}}\}\times\mathcal{G}^{\pm1}_{n_{L-1}}\times\ldots\times\mathcal{G}^{\pm1}_{n_1}\times\{\mathrm{id}_{\mathcal{G}_{n_0}}\}. \tag{49}$$

Since $\mathcal{G}^{>0}_{*}$ is the semidirect product of $\Delta^{>0}_{*}$ and $\mathcal{P}_{*}$ with $\Delta^{>0}_{*}$ is the normal subgroup, we will treat these two actions consecutively, $\Delta^{>0}_{*}$ first then $\mathcal{P}_{*}$. We denote these layers by $I_{\Delta^{>0}}$ and $I_{\mathcal{P}}$. Note that, since $I_{\Delta^{>0}}$ comes before $I_{\mathcal{P}}$, $I_{\Delta^{>0}}$ is required to be $\Delta^{>0}_{*}$-invariant and $\mathcal{P}_{*}$-equivariant, and $I_{\mathcal{P}}$ is required to be $\mathcal{P}_{*}$-invariant.

$\Delta^{>0}_{*}$-invariance and $\mathcal{P}_{*}$-equivariance. To capture $\Delta^{>0}_{*}$-invariance, we recall the notion of positively homogeneous of degree zero maps. For $n>0$, a map α from $\mathbb{R}^n$ is called *positively*

Table 9: Constraints of α component in invariant functional layer with $\mathrm{ReLU}, \sin, \mathrm{Tanh}$ activations.

Layer	$I_{\Delta^{>0}} \colon (W,b) \longmapsto (W',b')$	
	$\alpha^{(i)}_{jk} \ : \ W^{(i)}_{jk}) \longmapsto W'^{(i)}_{jk}$	$\alpha^{(i)}_{j} \ : \ b^{(i)}_{j} \longmapsto b'^{(i)}_{j}$
$i=1$	$\alpha^{(i)}_{\pi(j)k} = \alpha^{(i)}_{jk}$	$\alpha^{(i)}_{\pi(j)} = \alpha^{(i)}_{j}$
$1<i<L$	$\alpha^{(i)}_{\pi(j)\pi'(k)} = \alpha^{(i)}_{jk}$	$\alpha^{(i)}_{\pi(j)} = \alpha^{(i)}_{j}$
$i=L$	$\alpha^{(i)}_{j\pi(k)} = \alpha^{(i)}_{jk}$	$\alpha^{(i)}_{j}$

homogeneous of degree zero if

$$\alpha(\lambda x_1, \ldots, \lambda x_n) = \alpha(x_1, \ldots, x_n). \tag{50}$$

for all $\lambda > 0$ and $(x_1, \ldots, x_n) \in \mathbb{R}^n$. We construct $I_{\Delta^{>0}} \colon \mathcal{U} \to \mathcal{U}$ by taking collections of positively homogeneous of degree zero functions $\{\alpha^{(i)}_{jk} \colon \mathbb{R}^{w_i} \to \mathbb{R}^{w_i}\}$ and $\{\alpha^{(i)}_{j} \colon \mathbb{R}^{b_i} \to \mathbb{R}^{b_i}\}$, each one corresponds to weight and bias of $\mathcal{U}$. The maps $I_{\Delta^{>0}} \colon \mathcal{U} \to \mathcal{U}$ that $(W,b) \mapsto (W',b')$ is defined by simply applying these functions on each weight and bias entries as follows:

$$W'^{(i)}_{jk} = \alpha^{(i)}_{jk}(W^{(i)}_{jk}) \text{ and } b'^{(i)}_{j} = \alpha^{(i)}_{j}(b^{(i)}_{j}). \tag{51}$$

$I_{\Delta^{>0}}$ is $\Delta^{>0}_*$-invariant by homogeneity of the α functions. To make it become $\mathcal{P}_*$-equivariant, some α functions have to be shared arross any axis that have permutation symmetry, presented in Table 9.

Candidates of function α. We simply choose positively homogeneous of degree zero function $\alpha \ : \ \mathbb{R}^n \to \mathbb{R}^n$ by taking $\alpha(0) = 0$ and:

$$\alpha(x_1, \ldots, x_n) = \beta\left(\frac{x_1^2}{x_1^2 + \ldots + x_n^2}, \ldots, \frac{x_n^2}{x_1^2 + \ldots + x_n^2}\right). \tag{52}$$

where $\beta \colon \mathbb{R}^n \to \mathbb{R}^n$ is an arbitrary function. The function β can be fixed or parameterized to make α to be fixed or learnable.

$\mathcal{P}_*$-invariance. To capture $\mathcal{P}_*$-invariance, we simply take summing or averaging the weight and bias across any axis that have permutation symmetry as in [71]. In concrete, some $d > 0$, we have $I_{\mathcal{P}} \colon \mathcal{U} \to \mathbb{R}^d$ is computed as follows:

$$I_{\mathcal{P}}(U) = \left(W^{(1)}_{\star,:}, W^{(L)}_{:,\star}, W^{(2)}_{\star,\star}, \ldots, W^{(L-1)}_{\star,\star}; v^{(L)}, v^{(1)}_{\star}, \ldots, v^{(L-1)}_{\star}\right). \tag{53}$$

Here, $\star$ denotes summation or averaging over the rows or columns of the weight and bias.

$G-$invariance. Now we simply compose $I_{\mathcal{P}} \circ I_{\Delta^{>0}}$ to get an G-invariant map. We use an MLP to complete constructing an G-invariant layer with output dimension d as desired:

$$I = \text{MLP} \circ I_{\mathcal{P}} \circ I_{\Delta^{>0}}. \tag{54}$$

B.2 Sin or Tanh activation

Recall that, in this case:

$$G \coloneqq \{\mathrm{id}_{\mathcal{G}_{n_L}}\} \times \mathcal{G}^{\pm 1}_{n_{L-1}} \times \ldots \times \mathcal{G}^{\pm 1}_{n_1} \times \{\mathrm{id}_{\mathcal{G}_{n_0}}\}. \tag{55}$$

Since $\mathcal{G}^{\pm 1}_*$ is the semidirect product of $\Delta^{\pm 1}_*$ and $\mathcal{P}_*$ with $\Delta^{\pm 1}_*$ is the normal subgroup, we will treat these two actions consecutively, $\Delta^{\pm 1}_*$ first then $\mathcal{P}_*$. We denote these layers by $I_{\Delta^{\pm 1}}$ and $I_{\mathcal{P}}$. Note that, since $I_{\Delta^{\pm 1}}$ comes before $I_{\mathcal{P}}$, $I_{\Delta^{\pm 1}}$ is required to be $\Delta^{\pm 1}_*$-invariant and $\mathcal{P}_*$-equivariant, and $I_{\mathcal{P}}$ is required to be $\mathcal{P}_*$-invariant.

$\Delta_*^{\pm 1}$-invariance and $\mathcal{P}_*$-equivariance. To capture $\Delta_*^{\pm 1}$-invariance, we use even functions, i.e. $\alpha(x) = \alpha(-x)$ for all x. We construct $I_{\Delta^{\pm 1}} \colon \mathcal{U} \to \mathcal{U}$ by taking collections of even functions $\{\alpha_{jk}^{(i)} \colon \mathbb{R}^{w_i} \to \mathbb{R}^{w_i}\}$ and $\{\alpha_j^{(i)} \colon \mathbb{R}^{b_i} \to \mathbb{R}^{b_i}\}$, each one corresponds to weight and bias of $\mathcal{U}$. The maps $I_{\Delta^{\pm 1}} \colon \mathcal{U} \to \mathcal{U}$ that $(W, b) \mapsto (W', b')$ is defined by simply applying these functions on each weight and bias entries as follows:

$$W'^{(i)}_{jk} = \alpha_{jk}^{(i)}(W_{jk}^{(i)}) \text{ and } b'^{(i)}_j = \alpha_j^{(i)}(b_j^{(i)}). \tag{56}$$

$I_{\Delta^{\pm 1}}$ is $\Delta_*^{\pm 1}$-invariant by design. To make it become $\mathcal{P}_*$-equivariant, some α functions have to be shared arross any axis that have permutation symmetry, presented in Table 9.

Candidates of function α. We simply choose even function $\alpha \colon \mathbb{R}^n \to \mathbb{R}^n$ by:

$$\alpha(x_1, \ldots, x_n) = \beta\left(|x_1|, \ldots, |x_n|\right). \tag{57}$$

where $\beta \colon \mathbb{R}^n \to \mathbb{R}^n$ is an arbitrary function. The function β can be fixed or parameterized to make α to be fixed or learnable.

$\mathcal{P}_*$-invariance. To capture $\mathcal{P}_*$-invariance, we simply take summing or averaging the weight and bias across any axis that have permutation symmetry as in [71]. In concrete, some $d > 0$, we have $I_{\mathcal{P}} \colon \mathcal{U} \to \mathbb{R}^d$ is computed as follows:

$$I_{\mathcal{P}}(U) = \Big(W^{(1)}_{\star,:}, W^{(L)}_{:,\star}, W^{(2)}_{\star,\star}, \ldots, W^{(L-1)}_{\star,\star}; v^{(L)}, v^{(1)}_\star, \ldots, v^{(L-1)}_\star\Big). \tag{58}$$

Here, $\star$ denotes summation or averaging over the rows or columns of the weight and bias.

$G-$invariance. Now we simply compose $I_{\mathcal{P}} \circ I_{\Delta^{\pm 1}}$ to get an G-invariant map. We use an MLP to complete constructing an G-invariant layer with output dimension d as desired:

$$I = \text{MLP} \circ I_{\mathcal{P}} \circ I_{\Delta^{\pm 1}}. \tag{59}$$

C Proofs of Theoretical Results

C.1 Proof of Proposition 3.4

Proof. We simply denote the activation ReLU or $\sin$ or $\tanh$ by σ. Let $A \in \mathrm{GL}(n)$ that satisfies:

$$\sigma(A \cdot \mathbf{x}) = A \cdot \sigma(\mathbf{x}),$$

for all $\mathbf{x} \in \mathbb{R}^n$. This means:

$$\sigma\left(\begin{bmatrix} a_{11} & \ldots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \ldots & a_{nn} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}\right) = \begin{bmatrix} a_{11} & \ldots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \ldots & a_{nn} \end{bmatrix} \cdot \sigma\left(\begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}\right),$$

for all $x_1, \ldots, x_n \in \mathbb{R}$. We rewrite this equation as:

$$\sigma\left(\begin{bmatrix} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \\ \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n \end{bmatrix}\right) = \begin{bmatrix} a_{11} & \ldots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \ldots & a_{nn} \end{bmatrix} \cdot \begin{bmatrix} \sigma(x_1) \\ \vdots \\ \sigma(x_n) \end{bmatrix},$$

or equivalently:

$$\begin{bmatrix} \sigma(a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n) \\ \vdots \\ \sigma(a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n) \end{bmatrix} = \begin{bmatrix} a_{11}\sigma(x_1) + a_{12}\sigma(x_2) + \ldots + a_{1n}\sigma(x_n) \\ \vdots \\ a_{n1}\sigma(x_1) + a_{n2}\sigma(x_2) + \ldots + a_{nn}\sigma(x_n) \end{bmatrix}.$$

Thus,

$$\sigma\left(\sum_{j=1}^n a_{ij}x_j\right) = \sum_{j=1}^n a_{ij}\sigma(x_j),$$

for all $x_1, \ldots, x_n \in \mathbb{R}$ and $i = 1, \ldots, n$. We will consider the case $i = 1$, i.e.

$$\sigma\left(\sum_{j=1}^n a_{1j}x_j\right) = \sum_{j=1}^n a_{1j}\sigma(x_j), \tag{60}$$

and treat the case $i > 1$ similarly. Now we consider the activation σ case by case as follows.

(i) **Case 1.** $\sigma = \text{ReLU}$. We have some observations:

1. Let $x_1 = 1$, and $x_2 = \ldots = x_n = 0$. Then from Eq. (60), we have:
$$\sigma(a_{11}) = a_{11},$$
which implies that $a_{11} \geqslant 0$. Similarly, we also have $a_{12}, \ldots, a_{1n} \geqslant 0$.
2. Since A is an invertible matrix, the entries $a_{11}, \ldots, a_{1n}$ in the first row of A can not be simultaneously equal to 0.
3. There is at most only one nonzero number among the entries $a_{11}, \ldots, a_{1n}$. Indeed, assume by the contrary that $a_{11}, a_{12} > 0$. Let $x_3 = \ldots = x_n = 0$, from Eq. (60), we have:
$$\sigma(a_{11}x_1 + a_{12}x_2) = a_{11}\sigma(x_1) + a_{12}\sigma(x_2).$$
Let $x_2 = -1$, we have:
$$\sigma(a_{11}x_1 - a_{12}) = a_{11}\sigma(x_1).$$
Now, let $x_1 > 0$ be a sufficiently large number such that $a_{11}x_1 - a_{12} > 0$. (Note that this number exists since $a_{11} > 0$). Then we have:
$$a_{11}x_1 - a_{12} = a_{11}x_1,$$
which implies $a_{12} = 0$, a contradiction.

It follows from these three observations that there is exactly one non-zero element among the entries $a_{11}, \ldots, a_{1n}$. In other words, matrix A has exactly one nonzero entry in the first row. This applies for every row, so A has exactly one non-zero entry in each row. Since A is invertible, each column of A has at least one non-zero entry. Thus A also has exactly one non-zero entry in each column. Hence, A is in $\mathcal{G}_n$. Moreover, all entries of A are non-negative, so A is in $\mathcal{G}_n^{>0}$.

It is straight forward to check that for all A in $\mathcal{G}_n^{>0}$ we have $\sigma(A \cdot \mathbf{x}) = A \cdot \sigma(\mathbf{x})$.

(ii) **Case 2.** $\sigma = \text{Tanh}$ or $\sigma = \sin$. We have some observations:

1. Let $x_2 = \ldots = x_n = 0$. Then from Eq. (60), we have:
$$\sigma(a_{11}x_1) = a_{11}\sigma(x_1),$$
which implies $a_{11} \in \{-1, 0, 1\}$. Similarly, we have $a_{12}, \ldots, a_{1n} \in \{-1, 0, 1\}$.
2. Since A is an invertible matrix, the entries $a_{11}, \ldots, a_{1n}$ in the first row of A can not be simultaneously equal to 0.
3. There is at most only one nonzero number among the entries $a_{11}, \ldots, a_{1n}$. Indeed, assume by the contrary that $a_{11}, a_{12} \neq 0$. Let $x_3 = \ldots = x_n = 0$, from Eq. (60), we have:
$$\sigma(a_{11}x_1 + a_{12}x_2) = a_{11}\sigma(x_1) + a_{12}\sigma(x_2).$$
Note that $a_{11}, a_{12} \in \{-1, 1\}$, so by consider all the cases, we will lead to a contradiction.

It follows from the above three observations that there is exactly one non-zero element among the entries $a_{11}, \ldots, a_{1n}$. In other words, matrix A has exactly one nonzero entry in the first row. This applies for every row, so A has exactly one non-zero entry in each row. Note that, since A is invertible, each column of A has at least one non-zero entry. Therefore, A also has exactly one non-zero entry in each column. Hence, A is in $\mathcal{G}_n$. Moreover, all entries of A are in $\{-1, 0, 1\}$, so A is in $\mathcal{G}_n^{\pm 1}$.

It is straight forward to check that for all A in $\mathcal{G}_n^{\pm 1}$ we have $\sigma(A \cdot \mathbf{x}) = A \cdot \sigma(\mathbf{x})$.

The proposition is then proved completely. □

C.2 Proof of Proposition 4.4

Proof. For both Fully Connected Neural Networks case and Convolutional Neural Networks case, we consider a network f with three layers, with n_0, n_1, n_2, n_3 are number of channels at each layer, and its weight space $\mathcal{U}$. We will show the proof for part (i) where activation σ is ReLU, and part (ii) can be proved similarly. For part (i), we prove f to be G-invariant on its weight space $\mathcal{U}$, for the group G that is defined by:
$$G = \{\mathrm{id}_{\mathcal{G}_{n_3}}\} \times \mathcal{G}_{n_2}^{>0} \times \mathcal{G}_{n_1}^{>0} \times \{\mathrm{id}_{\mathcal{G}_{n_0}}\} < \mathcal{G}_{n_3} \times \mathcal{G}_{n_2} \times \mathcal{G}_{n_1} \times \mathcal{G}_{n_0} = \mathcal{G}_{\mathcal{U}};$$

Case 1. f is a Fully Connected Neural Network with three layers, with n_0, n_1, n_2, n_3 are number of channels at each layer as in Eq. 5:

$$f(\mathbf{x}\,;\,U,\sigma) = W^{(3)} \cdot \sigma\left(W^{(2)} \cdot \sigma\left(W^{(1)} \cdot \mathbf{x} + b^{(1)}\right) + b^{(2)}\right) + b^{(3)},$$

Case 2. f is a Convolutional Neural Network with three layers, with n_0, n_1, n_2, n_3 are number of channels at each layer as in Eq. 8:

$$f(\mathbf{x}\,;\,U,\sigma) = W^{(3)} * \sigma\left(W^{(2)} * \sigma\left(W^{(1)} * \mathbf{x} + b^{(1)}\right) + b^{(2)}\right) + b^{(3)}$$

We have some observations:

For case 1. For $W \in \mathbb{R}^{m\times n}, \mathbf{x} \in \mathbb{R}^n$ and $a > 0$, we have:

$$a \cdot \sigma(W \cdot \mathbf{x} + b) = \sigma\left((aW) \cdot \mathbf{x} + (ab)\right).$$

For case 2. For simplicity, we consider $*$ as one-dimentional convolutional operator, and other types of convolutions can be treated similarly. For $W = (w_1, \ldots, w_m) \in \mathbb{R}^m, b \in \mathbb{R}$ and $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$, we have:

$$W * \mathbf{x} + b = \mathbf{y} = (y_1, \ldots, y_{n-m+1}) \in \mathbb{R}^{n-m+1},$$

where:

$$y_i = \sum_{j=1}^{m} w_j x_{i+j-1} + b.$$

So for $a > 0$, we have:

$$a \cdot \sigma(W * \mathbf{x} + b) = \sigma\left((aW) * \mathbf{x} + (ab)\right).$$

With these two observations, we can see the proofs for both cases are similar to each other. We will show the proof for case 2, when f is a convolutional neural network since it is not trivial as case 1. Now we have $U = (W, b)$ with:

$$\begin{aligned} W &= \left(W^{(3)}, W^{(2)}, W^{(1)}\right), \\ b &= \left(b^{(3)}, b^{(2)}, b^{(1)}\right). \end{aligned}$$

Let g be an element of G:

$$g = \left(\mathrm{id}_{\mathcal{G}_{n_3}}, g^{(2)}, g^{(1)}, \mathrm{id}_{\mathcal{G}_{n_0}}\right),$$

where:

$$\begin{aligned} g^{(2)} &= D^{(2)} \cdot P_{\pi_2} = \operatorname{diag}\left(d^{(2)}_1, \ldots, d^{(2)}_{n_2}\right) \cdot P_{\pi_2} \in \mathcal{G}^{>0}_{n_2}, \\ g^{(1)} &= D^{(1)} \cdot P_{\pi_1} = \operatorname{diag}\left(d^{(1)}_1, \ldots, d^{(1)}_{n_1}\right) \cdot P_{\pi_1} \in \mathcal{G}^{>0}_{n_1}. \end{aligned}$$

We compute gU:

$$\begin{aligned} gU &= (gW, gb), \\ gW &= \left((gW)^{(3)}, (gW)^{(2)}, (gW)^{(1)}\right), \\ gb &= \left((gb)^{(3)}, (gb)^{(2)}, (gb)^{(1)}\right). \end{aligned}$$

where:

$$
\begin{aligned}
(gW)^{(3)}_{jk} &= \frac{1}{d^{(2)}_k} \cdot W^{(3)}_{j\pi_2^{-1}(k)},\\
(gW)^{(2)}_{jk} &= \frac{d^{(2)}_j}{d^{(1)}_k} \cdot W^{(2)}_{\pi_2^{-1}(j)\pi_1^{-1}(k)},\\
(gW)^{(1)}_{jk} &= \frac{d^{(1)}_j}{1} \cdot W^{(1)}_{\pi_1^{-1}(j)k},
\end{aligned}
$$

and,

$$
\begin{aligned}
(gb)^{(3)}_j &= b^{(3)}_j,\\
(gb)^{(2)}_j &= d^{(2)}_j \cdot b^{(2)}_{\pi_2^{-1}(j)},\\
(gb)^{(1)}_j &= d^{(1)}_j \cdot b^{(1)}_{\pi_1^{-1}(j)}.
\end{aligned}
$$

Now we show that $f(\mathbf{x}\ ;\ U,\sigma) = f(\mathbf{x}\ ;\ gU,\sigma)$ for all $\mathbf{x} = (x_1,\ldots,x_{n_0}) \in \mathbb{R}^{n_0}$. For $1 \leqslant i \leqslant n_3$, we compute the i-th entry of $f(\mathbf{x}\ ;\ gU,\sigma)$ as follows:

$$
\begin{aligned}
&f(\mathbf{x}\ ;\ gU,\sigma)_i\\
&= \sum_{j_2=1}^{n_2} (gW)^{(3)}_{ij_2} * \sigma\left(\sum_{j_1=1}^{n_1} (gW)^{(2)}_{j_2j_1} * \right.\\
&\qquad\qquad \left.\sigma\left(\sum_{j_0=1}^{n_0} (gW)^{(1)}_{j_1j_0} * x_{j_0} + (gb)^{(1)}_{j_1}\right) + (gb)^{(2)}_{j_2}\right) + (gb)^{(3)}_i\\
&= \sum_{j_2=1}^{n_2} \frac{1}{d^{(2)}_{j_2}} \cdot W^{(3)}_{i\pi_2^{-1}(j_2)} * \sigma\left(\sum_{j_1=1}^{n_1} \frac{d^{(2)}_{j_2}}{d^{(1)}_{j_1}} \cdot W^{(2)}_{\pi_2^{-1}(j_2)\pi_1^{-1}(j_1)} * \right.\\
&\qquad\qquad \left.\sigma\left(\sum_{j_0=1}^{n_0} \frac{d^{(1)}_{j_1}}{1} \cdot W^{(1)}_{\pi_1^{-1}(j_1)j_0} * x_{j_0} + d^{(1)}_{j_1} \cdot b^{(1)}_{\pi_1^{-1}(j_1)}\right) + d^{(2)}_{j_2} \cdot b^{(2)}_{\pi_2^{-1}(j_2)}\right) + b^{(3)}_i\\
&= \sum_{j_2=1}^{n_2} \frac{1}{d^{(2)}_{j_2}} \cdot W^{(3)}_{i\pi_2^{-1}(j_2)} * \sigma\left(\sum_{j_1=1}^{n_1} \frac{d^{(2)}_{j_2}}{d^{(1)}_{j_1}} \cdot W^{(2)}_{\pi_2^{-1}(j_2)\pi_1^{-1}(j_1)} * \right.\\
&\qquad\qquad \left.\sigma\left(d^{(1)}_{j_1} \cdot \left(\sum_{j_0=1}^{n_0} W^{(1)}_{\pi_1^{-1}(j_1)j_0} * x_{j_0} + b^{(1)}_{\pi_1^{-1}(j_1)}\right)\right) + d^{(2)}_{j_2} \cdot b^{(2)}_{\pi_2^{-1}(j_2)}\right) + b^{(3)}_i\\
&= \sum_{j_2=1}^{n_2} \frac{1}{d^{(2)}_{j_2}} \cdot W^{(3)}_{i\pi_2^{-1}(j_2)} * \sigma\left(\sum_{j_1=1}^{n_1} \frac{d^{(2)}_{j_2}}{d^{(1)}_{j_1}} \cdot W^{(2)}_{\pi_2^{-1}(j_2)\pi_1^{-1}(j_1)} * \right.\\
&\qquad\qquad \left.d^{(1)}_{j_1} \cdot \sigma\left(\sum_{j_0=1}^{n_0} W^{(1)}_{\pi_1^{-1}(j_1)j_0} * x_{j_0} + b^{(1)}_{\pi_1^{-1}(j_1)}\right) + d^{(2)}_{j_2} \cdot b^{(2)}_{\pi_2^{-1}(j_2)}\right) + b^{(3)}_i\\
&= \sum_{j_2=1}^{n_2} \frac{1}{d^{(2)}_{j_2}} \cdot W^{(3)}_{i\pi_2^{-1}(j_2)} * \sigma\left(\sum_{j_1=1}^{n_1} d^{(2)}_{j_2} \cdot W^{(2)}_{\pi_2^{-1}(j_2)\pi_1^{-1}(j_1)} * \right.\\
&\qquad\qquad \left.\sigma\left(\sum_{j_0=1}^{n_0} W^{(1)}_{\pi_1^{-1}(j_1)j_0} * x_{j_0} + b^{(1)}_{\pi_1^{-1}(j_1)}\right) + d^{(2)}_{j_2} \cdot b^{(2)}_{\pi_2^{-1}(j_2)}\right) + b^{(3)}_i
\end{aligned}
$$

$$
\begin{aligned}
&= \sum_{j_2=1}^{n_2} \frac{1}{d^{(2)}_{j_2}} \cdot W^{(3)}_{i\pi_2^{-1}(j_2)} * \sigma \Bigg(d^{(2)}_{j_2} \cdot \Bigg(\sum_{j_1=1}^{n_1} W^{(2)}_{\pi_2^{-1}(j_2)\pi_1^{-1}(j_1)} * \\
&\qquad\qquad \sigma \Bigg(\sum_{j_0=1}^{n_0} W^{(1)}_{\pi_1^{-1}(j_1)j_0} * x_{j_0} + b^{(1)}_{\pi_1^{-1}(j_1)} \Bigg) + b^{(2)}_{\pi_2^{-1}(j_2)} \Bigg) \Bigg) + b^{(3)}_i \\
&= \sum_{j_2=1}^{n_2} \frac{1}{d^{(2)}_{j_2}} \cdot W^{(3)}_{i\pi_2^{-1}(j_2)} \cdot d^{(2)}_{j_2} * \sigma \Bigg(\sum_{j_1=1}^{n_1} W^{(2)}_{\pi_2^{-1}(j_2)\pi_1^{-1}(j_1)} * \\
&\qquad\qquad \sigma \Bigg(\sum_{j_0=1}^{n_0} W^{(1)}_{\pi_1^{-1}(j_1)j_0} * x_{j_0} + b^{(1)}_{\pi_1^{-1}(j_1)} \Bigg) + b^{(2)}_{\pi_2^{-1}(j_2)} \Bigg) + b^{(3)}_i \\
&= \sum_{j_2=1}^{n_2} W^{(3)}_{i\pi_2^{-1}(j_2)} * \sigma \Bigg(\sum_{j_1=1}^{n_1} W^{(2)}_{\pi_2^{-1}(j_2)\pi_1^{-1}(j_1)} * \\
&\qquad\qquad \sigma \Bigg(\sum_{j_0=1}^{n_0} W^{(1)}_{\pi_1^{-1}(j_1)j_0} * x_{j_0} + b^{(1)}_{\pi_1^{-1}(j_1)} \Bigg) + b^{(2)}_{\pi_2^{-1}(j_2)} \Bigg) + b^{(3)}_i \\
&= \sum_{j_2=1}^{n_2} W^{(3)}_{ij_2} * \sigma \Bigg(\sum_{j_1=1}^{n_1} W^{(2)}_{j_2j_1} * \sigma \Bigg(\sum_{j_0=1}^{n_0} W^{(1)}_{j_1j_0} * x_{j_0} + b^{(1)}_{j_1} \Bigg) + b^{(2)}_{j_2} \Bigg) + b^{(3)}_i \\
&= f(\mathbf{x}\,;\, U, \sigma)_i.
\end{aligned}
$$

End of proof. □

D Additional experimental details

D.1 Runtime and Memory Consumption

We provide the runtime and memory consumption of Monomial-NFNs and the previous NFNs in Tables 10 and 11 to compare the computational and memory costs in the task of predicting CNN generalization (see Section 6.1). It is observable that our model runs faster and consumes significantly less memory than NP/HNP in [71] and GNN-based method in [35]. This highlights the benefits of parameter savings in Monomial-NFN.

Table 10: Runtime of models.

	NP [71]	HNP [71]	GNN [35]	Monomial-NFN (ours)
Tanh subset	35m34s	29m37s	4h25m17s	**18m23s**
ReLU subset	36m40s	30m06s	4h27m29s	**23m47s**

Table 11: Memory consumption.

	NP [71]	HNP [71]	GNN [35]	Monomial-NFN (ours)
Tanh subset	838MB	856MB	6390MB	**582MB**
ReLU subset	838MB	856MB	6390MB	**560MB**

D.2 Comparison of Monomial-NFNs and GNN-based NFNs

We provide experimental result to compare the efficiency of our model and a permutation equivariant GNN-based NFN [35] in two scenarios below.

1. Training the model on augmented train data and testing with the augmented test data (see Tables 12 and 13).

   Here, we present the experimental results on the original dataset and the results on the augmented dataset. The augmentation levels for the ReLU subset are 1, 2, 3, and 4, corresponding to augmentation ranges of $[1, 10], [1, 10^2], [1, 10^3], [1, 10^4]$. The augmented dataset for the Tanh subset corresponds to the augmentation range of $[-1, 1]$

Table 12: Predict CNN generalization on ReLU subset (augmented train data)

	Original	1	2	3	4
GNN [35]	0.897	0.892	0.885	0.858	0.851
Monomial-NF (ours)	**0.922**	**0.920**	**0.919**	**0.920**	**0.920**

Table 13: Predict CNN generalization on Tanh subset (augmented train data)

	Original	Augmented
GNN [35]	0.893	0.902
Monomial-NFN (ours)	**0.939**	**0.943**

The results for GNN exhibit a similar trend as other baselines that do not incorporate the scaling symmetry into their architectures. In contrast, our model has stable performance. A notable observation is that the GNN model uses 5.5M parameters (4 times more than our model), occupies 6000MB of memory, and takes 4 hours to train.

2. Training the model on original train data and testing with the augmented test data (see Tables 14 and 15).

Table 14: Predict CNN generalization on ReLU subset (original train data)

Augment level	1	2	3	4
GNN [35]	0.794	0.679	0.586	0.562
Monomial-NF (ours)	**0.920**	**0.919**	**0.920**	**0.920**

Table 15: Predict CNN generalization on Tanh subset (original train data)

	Augmented
GNN [35]	0.883
Monomial-NFN (ours)	**0.940**

In these more challenging scenario, GNN's performance drops significantly, which highlights the lack of scaling symmetry in the model. Our model maintains consistent performance, matching the case in which we train with the augmented data.

D.3 Predicting generalization from weights

Dataset. The original ReLU subset of the CNN Zoo dataset includes 6050 instances for training and 1513 instances for testing. For the Tanh dataset, it includes 5949 training and 1488 testing instances. For the augmented data, we set the augmentation factor to 2, which means that we augment the original data once, resulting in a new dataset of double the size. The complete size of all datasets is presented in Table 16

Implementation details. Our model follows the same architecture as in [71], comprising three equivariant Monomial-NFN layers with 16, 16, and 5 channels, respectively, each followed by ReLU activation (ReLU dataset) or Tanh activation (Tanh dataset). The resulting weight space features are input into an invariant Monomial-NFN layer with Monomial-NFN pooling (Equation 19) with learnable parameters (ReLU case) or mean pooling (Tanh case). Specifically, the Monomial-NFN pooling layer normalizes the weights across the hidden dimension and takes the average for rows (first layer), columns (last layer), or both (other layers). The output of this invariant Monomial-NFN layer is flattened and projected to $\mathbb{R}^{200}$ (ReLU case) or $\mathbb{R}^{1000}$ (Tanh case). This resulting vector is then passed through an MLP with two hidden layers with ReLU activations. The output is linearly projected to a scalar and then passed through a sigmoid function. We use the Binary Cross Entropy (BCE) loss function and train the model for 50 epochs, with early stopping based on τ on the validation set, which takes 35 minutes to train on an A100 GPU. The hyperparameters for our model are presented in Table 18.

Table 16: Datasets information for predicting generalization task.

Dataset	Train size	Val size
Original ReLU	6050	1513
Original Tanh	5949	1488
Augment ReLU	12100	3026
Augment Tanh	11898	2976

Table 17: Number of parameters of all models for prediciting generalization task.

Model	ReLU dataset	Tanh dataset
STATNN	1.06M	1.06M
NP	2.03M	2.03M
HNP	2.81M	2.81M
Monomial-NFN (ours)	0.25M	1.41M

Table 18: Hyperparameters for Monomial-NFN on prediciting generalization task.

	ReLU	Tanh
MLP hidden neurons	200	1000
Loss	Binary cross-entropy	Binary cross-entropy
Optimizer	Adam	Adam
Learning rate	0.001	0.001
Batch size	8	8
Epoch	50	50

Table 19: Dataset size for Classifying INRs task.

	Train	Validation	Test
CIFAR-10	45000	5000	10000
MNIST size	45000	5000	10000
Fashion-MNIST	45000	5000	20000

For the baseline models, we follow the original implementations described in [71], using the official code (available at: https://github.com/AllanYangZhou/nfn). For the HNP and NP models, there are 3 equivariant layers with 16, 16, and 5 channels, respectively. The features go through an average pooling layer and 3 MLP layers with 1000 hidden neurons. The hyperparameters of our model and the number of parameters for all models in this task can be found in Table 17.

D.4 Classifying implicit neural representations of images

Dataset. We utilize the original INRs dataset provided by [71], with no augmentation. The data is obtained by implementing a single SIREN model for each image in each dataset: CIFAR-10, MNIST, and Fashion-MNIST. The size of training, validation, and test samples for each dataset is provided in Table 19.

Implementation details. In these experiments, our general architecture includes 2 Monomial-NFN layers with sine activation, followed by 1 Monomial-NFN layer with absolute activation. The choice of hidden dimension in the Monomial-NFN layer depends on each dataset and is described in Table 20. The architecture then follows the same design as the NP and HNP models in [71], where a Gaussian Fourier Transformation is applied to encode the input with sine and cosine components, mapping from 1 dimension to 256 dimensions. If the base layer is NP, the features will go through IOSinusoidalEncoding, a positional encoding designed for the NP layer, with a maximum frequency of 10 and 6 frequency bands. After that, the features go through 3 HNP or NP layers with ReLU activation functions. Then, an average pooling is applied, and the output is flattened, and the resulting vector is passed through an MLP with two hidden layers, each containing 1000 units and ReLU activations. Finally, the output is linearly projected to a scalar. For the MNIST dataset, there is an additional Channel Dropout layer after the ReLU activation of each HNP layer and a Dropout layer after the ReLU activation of each MLP layer, both with a dropout rate of 0.1. We use the Binary Cross Entropy (BCE) loss function and train the model for 200,000 steps, which takes 1 hour and 35

Table 20: Hyperparameters of Monomial-NFN for each dataset in Classify INRs task.

	MNIST	Fashion-MNIST	CIFAR-10
Monomial-NFN hidden dimension	64	64	16
Base model	HNP	NP	HNP
Base model hidden dimension	256	256	256
MLP hidden neurons	1000	500	1000
Dropout	0.1	0	0
Learning rate	0.000075	0.0001	0.0001
Batch size	32	32	32
Step	200000	200000	200000
Loss	Binary cross-entropy	Binary cross-entropy	Binary cross-entropy

Table 21: Number of parameters of all models for classifying INRs task.

	CIFAR-10	MNIST	Fashion-MNIST
MLP	2M	2M	2M
NP	16M	15M	15M
HNP	42M	22M	22M
Monomial-NFN (ours)	16M	22M	20M

Table 22: Number of parameters of all models for Weight space style editing task.

Model	Number of parameters
MLP	4.5M
NP	4.1M
HNP	12.8M
Monomial-NFN (ours)	4.1M

Table 23: Hyperparameters for Monomial-NFN on weight space style editing task.

Name	Value
Monomial-NFN hidden dimension	16
NP dimension	128
Optimizer	Adam
Learning rate	0.001
Batch size	32
Steps	50000

minutes on an A100 GPU. For the baseline models, we follow the same architecture in [71], with minor modifications to the model hidden dimension, reducing it from 512 to 256 to avoid overfitting. We use a hidden dimension of 256 for all baseline models and our base model. The number of parameters of all models can be found in Table 21

D.5 Weight space style editing

Dataset. We use the same INRs dataset as used for classification task, which has the size of train, validation and test set described in Table 19.

Implementation details. In these experiments, our general architecture includes 2 Monomial-NFN layers with 16 hidden dimensions. The architecture then follows the same design as the NP model in [71], where a Gaussian Fourier Transformation with a mapping size of 256 is applied. After that, the features go through IOSinusoidalEncoding and then through 3 NP layers, each with 128 hidden dimensions and ReLU activation. Finally, the output goes through an NP layer to project into a scalar and a LearnedScale layer described in the Appendix of [71]. We use the Binary Cross Entropy (BCE) loss function and train the model for 50,000 steps, which takes 35 minutes on an A100 GPU. For the baseline models, we keep the same settings as the official implementation. Specifically, the HNP or NP model will have 3 layers, each with 128 hidden dimensions, followed by a ReLU activation. An NFN of the same type will be applied to map the output to 1 dimension and pass it through a LearnedScale layer. The number of parameters of all models can be found in Table 22. The detailed hyperparameters for our model can be found in Table 23.

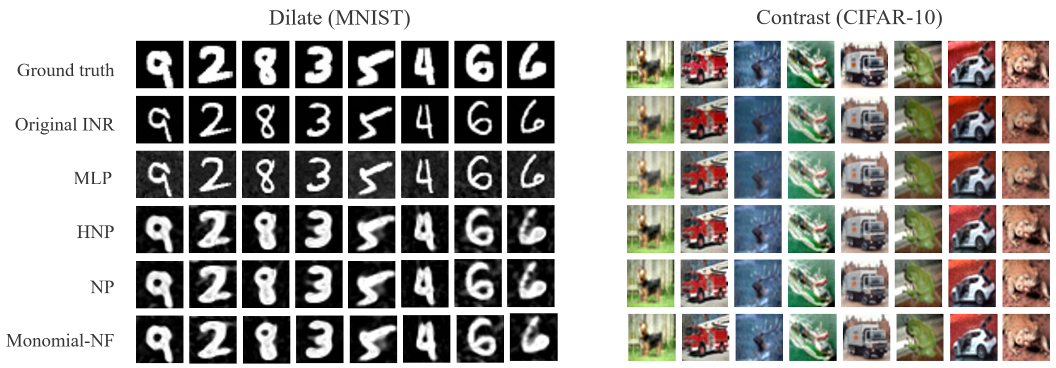

Figure 2: Random qualitative samples of INR editing behavior on the Dilate (MNIST) and Contrast (CIFAR-10) editing tasks.

D.6 Ablation Regarding Design Choices

We provide the ablation study on the choice of architecture for the task Predict CNN Generalization on ReLU subset in Table 24. We denote:

- Monomial Equivariant Functional Layer (Ours): MNF
- Activation: ReLU
- Scaling Invariant and Permutation Equivariant Layer (Ours): Norm
- Hidden Neuron Permutation Invariant Layer (in [71]): HNP
- Permutation Invariant Layer: Avg
- Multilayer Perceptron: MLP

Table 24: Ablation study on design choices for the task Predict CNN generalization on ReLU subset

	Original	1	2	3	4
(MNF–ReLU)$\times$1 $\rightarrow$ Norm $\rightarrow$ (HNP–ReLU)$\times$1 $\rightarrow$ Avg $\rightarrow$ MLP	0.917	0.916	0.917	0.917	0.917
(MNF–ReLU)$\times$2 $\rightarrow$ Norm $\rightarrow$ (HNP–ReLU)$\times$1 $\rightarrow$ Avg $\rightarrow$ MLP	0.918	0.917	0.917	0.917	0.918
(MNF–ReLU)$\times$3 $\rightarrow$ Norm $\rightarrow$ (HNP–ReLU)$\times$1 $\rightarrow$ Avg $\rightarrow$ MLP	0.920	0.919	0.918	0.920	0.920
(MNF–ReLU)$\times$1 $\rightarrow$ Norm $\rightarrow$ Avg $\rightarrow$ MLP	0.915	0.914	0.917	0.916	0.914
(MNF–ReLU)$\times$2 $\rightarrow$ Norm $\rightarrow$ Avg $\rightarrow$ MLP	0.918	0.919	0.918	0.917	0.918
(MNF–ReLU)$\times$3 $\rightarrow$ Norm $\rightarrow$ Avg $\rightarrow$ MLP	**0.922**	**0.920**	**0.919**	**0.920**	**0.920**

Among these designs, the architecture incorporating three layers of Monomial-NFN with ReLU activation achieves the best performance.