Journal of Machine Learning Research 24 (2023) 1-40 Submitted 3/23; Revised 10/23; Published 11/23

Densely Connected G-invariant Deep Neural Networks with Signed Permutation Representations

Devanshu Agrawal DAGRAWA2@VOLS.UTK.EDU
Department of Industrial and Systems Engineering
University of Tennessee
Knoxville, TN 37996, USA

James Ostrowski JOSTROWS@UTK.EDU
Department of Industrial and Systems Engineering
University of Tennessee
Knoxville, TN 37996, USA

Editor: Joan Bruna

Abstract

We introduce and investigate, for finite groups G, G-invariant deep neural network (G-DNN) architectures with ReLU activation that are densely connected– i.e., include all possible skip connections. In contrast to other G-invariant architectures in the literature, the preactivations of the$G$-DNNs presented here are able to transform by *signed* permutation representations (signed perm-reps) of G. Moreover, the individual layers of the G-DNNs are not required to be G-equivariant; instead, the preactivations are constrained to be G-equivariant functions of the network input in a way that couples weights across all layers. The result is a richer family of G-invariant architectures never seen previously. We derive an efficient implementation of G-DNNs after a reparameterization of weights, as well as necessary and sufficient conditions for an architecture to be "admissible"– i.e., non-degenerate and inequivalent to smaller architectures. We include code that allows a user to build a G-DNN interactively layer-by-layer, with the final architecture guaranteed to be admissible. We show that there are far more admissible G-DNN architectures than those accessible with the "concatenated ReLU" activation function from the literature. Finally, we apply G-DNNs to two example problems—(1) multiplication in $\{-1, 1\}$ (with theoretical guarantees) and (2) 3D object classification—finding that the inclusion of signed perm-reps significantly boosts predictive performance compared to baselines with only ordinary (i.e., unsigned) perm-reps.

Keywords: deep learning, group theory, neural network, skip connection, symmetry

1. Introduction

When fitting a deep neural network (DNN) to a target function that is known to be G-invariant with respect to a group G, it only makes sense to enforce G-invariance on the DNN as prior knowledge. With the rise of geometric deep learning (Bronstein et al., 2021), this is becoming an increasingly common practice, finding applications in various domains including computer vision (Veeling et al., 2018; Esteves et al., 2018; Musallam et al., 2022), the physical sciences (Luo et al., 2021; Atz et al., 2021; Kaba and Ravanbakhsh, 2022; Agrawal et al., 2023), and genomics (Mallet and Vert, 2021; Zhou et al., 2022). In general-

purpose G-invariant and G-equivariant architectures such as G-equivariant convolutional neural networks (G-CNNs) (Cohen and Welling, 2016) and G-equivariant graph neural networks (Maron et al., 2019a), it is standard to require every linear layer to be G-equivariant. Moreover, in case of the rectified linear unit (ReLU) activation, every linear layer is G-equivariant only with respect to permutation representations. It is commonly assumed that the G-invariant architectures constructed in this way are sufficient for consideration (Cohen et al., 2019; Finzi et al., 2021), but it is unclear if this layerwise construction covers all possible ways of enforcing G-invariance on a fully-connected feedforward DNN, and it remains an open conjecture (Kondor and Trivedi, 2018). While these architectures and others are certainly sufficient for universal approximation of G-invariant functions (Maron et al., 2019b; Ravanbakhsh, 2020; Kicki et al., 2020), the way in which G-invariance is enforced is an aspect of the neural architecture and thus likely plays a key role in determining the inductive bias of the model and hence its generalization power on a given problem.

It has recently been discovered that there are, in fact, more ways to enforce G-invariance on shallow ReLU neural networks than just permutations on the hidden neurons (Agrawal and Ostrowski, 2022). In their work, Agrawal and Ostrowski (2022) exploit the identity

$$\mathrm{ReLU}(-x) = \mathrm{ReLU}(x) - x \tag{1}$$

to show that G-invariance can be achieved even if G acts on the hidden neurons via a "signed permutation representation" (signed perm-rep), resulting in novel G-invariant shallow architectures previously unknown. In an attempt towards a generalization to deep architectures, we observe that the linear term in Eq. (1) can be interpreted as a skip connection; this suggests that skip connections may be the key to novel deep G-invariant architectures.

In this paper, as a partial generalization of the work of Agrawal and Ostrowski (2022), we investigate G-invariant deep neural network (G-DNN) architectures that are "densely connected"– i.e., include all possible skip connections. We note that (non-G-invariant) densely connected neural networks exist in the literature and have found immense success especially in medical imaging (Huang et al., 2017). We use ReLU activation, and every preactivation layer is still a G-equivariant function of the network input; however, in contrast to previous architectures such as the G-CNN, the individual weight matrices of a G-DNN need not be G-equivariant. Instead, in each layer, only the concatenation of the weight matrix with all skip connections from previous layers need be G-equivariant. This dense structure allows us to use *signed* perm-reps to enforce$G$-equivariance on the preactivation layers, thus granting us access to a much larger family of G-invariant architectures than seen previously.

Implementation of G-DNNs is nontrivial, as the group representation by which the weight matrix (concatenated with all skip connections) in each layer transforms is itself a function of the weights in previous layers. That is, due to the skip connections and in contrast to previous architectures such as the G-CNN, G-invariance is enforced in a way that couples weights across layers. We show, however, that there is a reparameterization of G-DNNs in which the equivariance conditions of the weight matrices decouple and admit a simple implementation (Thm. 6 (a)). For efficiency, rather than transforming back to the original weights, we express and implement the forward pass of a G-DNN directly in terms of these reparameterized weights (Thm. 6 (b)).

There is additional literature suggesting the potential of G-DNNs. Signed perm-reps have been used previously as design components in G-invariant architectures and have proved beneficial especially in computer vision (Cohen and Welling, 2017). These previous works incorporate signed perm-reps by replacing ReLU with the *concatenated ReLU* (CRELU) activation function (Shang et al., 2016) defined as

$$\mathrm{CReLU}(x) = \mathrm{ReLU}\left(\begin{bmatrix} x \\ -x \end{bmatrix}\right), \tag{2}$$

which sends negations of its input to transpositions of its output. CReLU was originally introduced to capture the empirical observation that certain feature maps in convolutional neural networks trained on images tend to pair up (Shang et al., 2016). Since then, besides its use in G-invariant architectures, CReLU has been shown to be beneficial for the trainability of networks with skip connections– e.g., its compatibility with batch normalization (Shang et al., 2017) and its role in the "looks linear" weight initialization (Balduzzi et al., 2017). The CReLU literature thus suggests that G-invariance, signed perm-reps, and skip connections may play well together. and we claim that the G-DNN architectures introduced in this paper are a culmination of these ideas. In contrast to the literature, however, our perspective is top-down, and we show in this paper that a systematic study of G-DNN architectures leads to the discovery of novel architecture designs not accessible with CReLU alone (Ex. 1).

Finally, this work contributes to the ultimate goal of G-invariant neural architecture search (G-NAS), which purports to apply search methods over G-invariant architectures. In addition to finding good G-invariant architectures for a given problem, G-NAS would relieve practitioners from having to learn or perform specialized mathematics (in particular, group theory), thereby making G-invariant deep learning more accessible and applicable. Previous works have already begun to explore G-NAS with promising results (Basu et al., 2021; Maile et al., 2022), but these works only operate within a limited region of G-invariant architecture space; indeed, Agrawal and Ostrowski (2022) argue that a more extensive G-NAS first requires the characterization of all G-invariant architectures as well as the so-called network morphisms between them. There has been work characterizing G-invariant architectures from a graph-theoretic perspective but not exploiting identities of the activation function (Ravanbakhsh et al., 2017); work on the classification of G-invariant shallow ReLU architectures (Agrawal and Ostrowski, 2022); and even work introducing novel G-invariant architectures based on a sum-product layer (Kicki et al., 2021). Now, our discovery of novel G-DNN architectures based on Eq. (1) pushes the horizon on G-invariant architecture space further back, revealing new regions of exploration. We prove Thm. 7, which lets us count numbers of distinct non-degenerate G-DNNs, in terms of which we gain intuition about how many new architectures we are now able to access.

The remainder of the paper is organized as follows: In Sec. 2, we review signed perm-reps and state our central hypothesis with some theoretical support. In Sec. 3, we introduce and describe the implementation of G-DNN architectures. We additionally derive necessary and sufficient conditions for a G-DNN architecture to be "admissible" or nondegenerate (Thm. 7), in the sense that (1) no neuron is missing an input and (2) no two ReLU neurons

can be combined into a single ReLU neuron or a skip connection. We include code[1] that allows a user to build a G-DNN interactively such that the final architecture is guaranteed to be admissible. We also verify that batch normalization placed after ReLU is compatible with G-DNNs out-of-the-box. In Sec. 4, we demonstrate that G-DNNs go well-beyond CReLU-based architectures, and we test G-DNNs on two examples—(1) a simple mathematical function and (2) a real-world computer vision problem—and demonstrate that signed perm-reps can, in fact, carry useful inductive bias. Finally, in Sec. 5, we end with conclusions, implications, and future outlook.

2. Signed permutation representations

2.1 Preliminaries

We begin by introducing some notions and notation that will be used throughout this paper. This section is an abbreviation of Secs. 2.1-2.2 of Agrawal and Ostrowski (2022), and we refer readers to all of Sec. 2 of that paper for details of the below material.

Throughout this paper, let G be a finite group of $m \times m$ orthogonal matrices. Let $\mathrm{P}(n)$ be the group of $n \times n$ permutation matrices and $\mathrm{Z}(n)$ the group of $n \times n$ diagonal matrices with diagonal entries ± 1. Let $\mathrm{PZ}(n)$ be the group of *signed permutations*– i.e., the group of all permutations and reflections of the standard orthonormal basis $\{e_1, \ldots, e_n\}$. For interested readers, this group is the semidirect product $\mathrm{P}(n) \ltimes \mathrm{Z}(n)$, and it is also called the hyperoctahedral group in the literature (Baake, 1984).

A *signed permutation representation* (signed perm-rep) of degree n of G is a homomorphism $\rho : G \mapsto \mathrm{PZ}(n)$. Two signed perm-reps ρ, ρ' are said to be *equivalent* or *conjugate* if there exists $A \in \mathrm{PZ}(n)$ such that $\rho'(g) = A\rho(g)A^{-1} \forall g \in G$– i.e., if they are related by a change of basis. A signed perm-rep ρ is said to be *reducible* if it is equivalent to a direct sum of signed perm-reps of smaller degrees– i.e., if there exists $A \in \mathrm{PZ}(n)$ such that $A\rho(\cdot)A^{-1}$ is simultaneously block-diagonal with at least two blocks. The signed perm-rep is said to be *irreducible* (signed perm-irrep for short) otherwise.[2]

The signed perm-irreps of G can be completely classified up to equivalence in terms of certain pairs of subgroups of G (Agrawal and Ostrowski, 2022, Thm. 1). For every pair of subgroups $K \leq H \leq G$, $|H : K| \leq 2$, define the signed perm-rep $\rho_{HK} : G \mapsto \mathrm{PZ}(n)$, $n = |G/H|$, to be the induced representation

$$\rho_{HK} = I_H^G \sigma, \text{ where } \sigma : H \mapsto \{-1, 1\} \mid \ker(\sigma) = K.$$

Then ρ_{HK} is irreducible, and every signed perm-irrep is equivalent to some ρ_{HK}. Moreover, ρ_{HK} and $\rho_{H'K'}$ are equivalent iff $(H', K') = (gHg^{-1}, gKg^{-1})$ for some $g \in G$. We can thus always understand a "signed perm-irrep" to mean ρ_{HK} for some appropriate subgroups H and K, and conversely we will always understand the notation ρ_{HK} to mean the above construction.

For a more explicit but equivalent construction of ρ_{HK}, let $\{g_1, \ldots, g_n\}$ be a transversal of G/H. If $|H : K| = 1$ (i.e., $H = K$), then we define $\rho_{HK}(g)e_i = e_j$ iff $gg_iH = g_jH$;

1. Code for our implementation and for reproducing all results in this paper is available at: `https://github.com/dagrawa2/gdnn_code`.
2. A useful characterization is that a signed perm-rep ρ is irreducible iff for every $i, j = 1, \ldots, n$, there exists $g \in G$ such that $\rho(g)e_i = \pm e_j$.

i.e., ρ_{HK} is just defined in terms of its permutation action on G/H. On the other hand, suppose $|H : K| = 2$. Then we have the quotient group $H/K = \{K, hK\}$. Define $e_{-i} = -e_i$ and $g_{-i} = g_i h$ for $i \in \{1, \dots, n\}$. Then we define $\rho_{HK}(g)e_i = e_j$ iff $gg_iK = g_jK$ for $i, j \in \{\pm 1, \dots, \pm n\}$. Intuitively, ρ_{HK} still acts on G/H via permutations; however, if the cosets in H/K are swapped under this action, then a sign flip is incurred in the representation. For more details on these constructions, see Sec. 2 of Agrawal and Ostrowski (2022).

Every signed perm-irrep is either type 1 or type 2. A signed perm-rep is *type 1* if it is equivalent to an *ordinary permutation representation* (ordinary perm-rep) $\pi : G \mapsto \mathrm{P}(n)$; it is *type 2* otherwise. Sign flips in a type 1 signed perm-rep are thus artifacts as they can be removed by a change of basis. An important characterization is that a signed perm-irrep ρ_{HK} has type $|H : K|$.

Finally, as general notation, throughout this paper let I_n denote the $n \times n$ identity matrix and $\vec{0}_n$ (resp. $\vec{1}_n$) the n-dimensional vector with all elements 0 (resp. 1).

2.2 Central hypothesis

Previous works in the equivariant deep learning literature, to our knowledge, primarily employ type 1 signed perm-reps—and in particular, ordinary perm-reps—to enforce layerwise G-equivariance in ReLU networks. The goal of this paper is to show that type 2 signed perm-reps can also be implemented for this purpose (although it is trickier as it requires layers to be coupled), and the central hypothesis of this work is that type 2 signed perm-reps can indeed be beneficial for DNN performance. In this section, we motivate this hypothesis with theory and discussions that suggest that type 2 signed perm-reps can help boost expressive power without necessarily sacrificing generalization power. We note that the CReLU activation function has previously been used to incorporate certain type 2 signed perm-reps (see Sec. 1), but we take a top-down approach that lets us study all such architectures.

Our first theorem states that two inequivalent signed perm-irreps induced from different reps of the same subgroup $H \leq G$ correspond to G-equivariant matrices that are in a sense orthogonal.[3]

Theorem 1 (Corollary to Prop. 6 of Agrawal and Ostrowski (2022)) *Let ρ_{HK_1} and ρ_{HK_2} be inequivalent signed perm-irreps of degree n of G. For each $i \in \{1, 2\}$, let $W_i \in \mathbb{R}^{n \times m}$ such that $\rho_{HK_i}(g)W_i = W_i g \forall g \in G$. Then there exists $P \in \mathrm{P}(n)$ such that*

$$\mathrm{diag}(PW_1W_2^\top) = 0.$$

Theorem 1 will be important in Sec. 4, where we will use it to construct a key baseline G-DNN architecture for our experiments. We discuss Thm. 1 further in the context of an example group G later in this section.

Our second theorem describes how a signed perm-rep can be "unraveled" into an ordinary perm-rep of twice the degree, by essentially mapping sign flips to transpositions of two dimensions. This unraveling is particularly meaningful for type 2 signed perm-irreps, which remain irreducible even after unraveling; in contrast, type 1 signed perm-irreps merely unravel into reducible copies of themselves. Let $\mathcal{H}$ be the elementwise Heaviside step function, where we let $\mathcal{H}(0) = 0$.

3. All proofs can be found in the supplementary material.

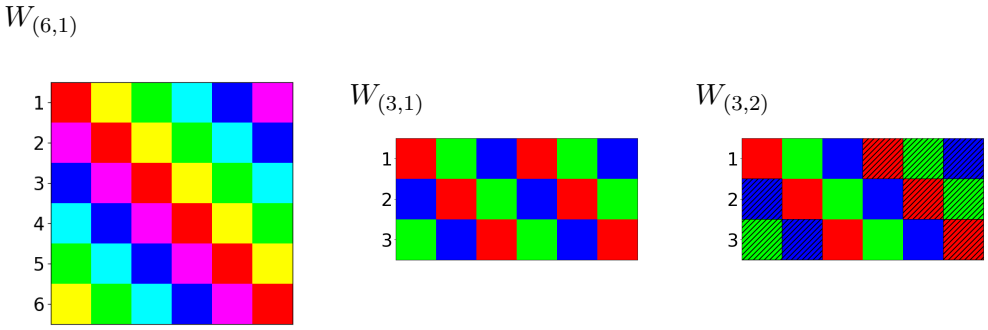

Figure 1: Weightsharing patterns of equivariant matrices for three example signed perm-irreps of the group G of 6×6 cyclic permutation matrices (see main text for details). In each pattern, weights of the same color and texture (solid vs. hatched) are constrained to be equal; weights of the same color but different texture are constrained to be opposites (colors should not be compared across different matrices).

Theorem 2 *Let $\rho : G \mapsto \mathrm{PZ}(n)$ be a signed perm-rep. Then:*

(a) The function[4]

$$\pi_\rho(g) = \mathcal{H}\left(\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \otimes \rho(g)\right)$$

defines an ordinary perm-rep $\pi_\rho : G \mapsto \mathrm{P}(2n)$.

(b) Let $W \in \mathbb{R}^{n \times m}$ such that $\rho(g)W = Wg \forall g \in G$. Then

$$\pi_\rho(g) \begin{bmatrix} W \\ -W \end{bmatrix} = \begin{bmatrix} W \\ -W \end{bmatrix} g \forall g \in G.$$

(c) Suppose ρ is irreducible. Then π_ρ is irreducible iff ρ is type 2.

(d) Suppose ρ is type 2 irreducible and $\rho = \rho_{HK}$. Then $\pi_\rho = \rho_{KK}$.

To understand Thms. 1-2 and their implications for the expressive and generalization powers of a G-DNN with type 2 signed perm-reps, consider the example group G of all 6×6 cyclic permutation matrices (see Sec. 4.1 of Agrawal and Ostrowski (2022) for a detailed discussion of this example). For this example, the signed perm-irreps of G are completely characterized in terms of their degrees and types; we thus let $\rho_{(n,t)}$ denote the irrep ρ_{HK} where $n = \frac{6}{|H|}$ is the degree and $t = |H : K|$ is the type. Alternatively, if $G \cong \mathbb{Z}_6$, then $\rho_{(n,t)}$ denotes the irrep ρ_{HK} where $H \cong \mathbb{Z}_{\frac{6}{n}}$ and $K \cong \mathbb{Z}_{\frac{6}{nt}}$. In this notation, the signed perm-irreps of G are $\rho_{(6,1)}$, $\rho_{(3,1)}$, $\rho_{(3,2)}$, $\rho_{(2,1)}$, $\rho_{(1,1)}$, and $\rho_{(1,2)}$.

For each irrep $\rho_{(n,t)}$, let $W_{(n,t)} \in \mathbb{R}^{n \times m}$ such that $\rho_{(n,t)} W_{(n,t)} = W_{(n,t)} g \forall g \in G$. The irreps $\rho_{(3,1)}$ and $\rho_{(3,2)}$ satisfy the hypotheses of Thm. 1, and indeed we see—in terms of their

4. In the expression for π_ρ, the operation $\otimes$ is the Kronecker product.

weightsharing patterns—each row of $W_{(3,1)}$ is orthogonal to the corresponding row of $W_{(3,2)}$ (Fig. 1). In practice, this means if we were given a dataset sampled from a G-invariant shallow neural network (G-SNN)

$$f(x) = a\vec{1}^\top \operatorname{ReLU}(Wx + b\vec{1}) + c^\top x + d,$$

with $W = W_{(3,2)}$ as the ground truth weight matrix, then a G-SNN with weight matrix with the weightsharing pattern of $W_{(3,1)}$ would not have the expressive power to fit the ground truth, regardless of the number of "channels" (i.e., independent copies of $W_{(3,1)}$) used. With a limited budget of three hidden neurons per channel, type 2 signed perm-reps thus help to increase expressive power.

On the other hand, if we double our budget of hidden neurons, then Thm. 2 suggests $W_{(6,1)}$ has the capacity to express $W_{(3,2)}$; indeed, if we constrain the first row of $W_{(6,1)}$ to have the same weightsharing pattern as the first row of $W_{(3,2)}$ (Fig. 1), then $W_{(6,1)}$ effectively becomes equivalent to $W_{(3,2)}$. The problem is, however, that $W_{(6,1)}$ can similarly be constrained to match $W_{(3,1)}$, and with probability 1 (e.g., under a Gaussian), $W_{(6,1)}$ is equivalent to neither $W_{(3,1)}$ nor $W_{(3,2)}$. Thus, while this approach allows us to use a type 1 signed perm-rep with the capacity to express the ground truth, it comes at the cost of generalization power, as $W_{(6,1)}$ may have multiple configurations consistent with a finite training set.

To ensure these concepts are made clear, we present one more very simple example. For any even input dimension $n \geq 2$, consider the G-SNN architectures

$$f_i(x) = \vec{1}^\top \operatorname{ReLU}(W_i x) - \frac{1}{2}\vec{1}^\top W_i x \forall i \in \{1, 2, 3\},$$

where

$$\begin{aligned} W_1 &= \begin{bmatrix} u & u & \cdots & u & u \end{bmatrix} \\ W_2 &= \begin{bmatrix} u & -u & \cdots & u & -u \end{bmatrix} \\ W_3 &= \begin{bmatrix} u & v & \cdots & u & v \end{bmatrix}, \end{aligned}$$

and where $\vec{1}$ is a vector of 1's of the appropriate length (either 1 or 2). Here G is the group of cyclic permutations on n elements. To fix a global scale, without loss of generality, let us set $u = 1$. Now suppose we take the type 2 architecture f_2 as the ground truth, and draw a sample dataset. We then observe that the type 1 architecture f_1 does not have the capacity to fit this dataset; in fact, f_1 and f_2 have orthogonal level sets. On the other hand, the unraveled architecture f_3 does have the capacity to fit the data, but it is a weaker model compared to f_2 (with respect to this dataset) as it must learn $v = -u$ from the data.

In sum, type 2 signed perm-reps may help to refine the range of expressive powers available to us; they help to express functions different from type 1 signed perm-reps of the same degree without sacrificing generalization power the way larger type 1 signed perm-reps would.

3. G-invariant deep neural networks

3.1 Parameterization redundancies

In this section, we define densely connected deep neural networks, or simply deep neural networks, and we list some of the so-called parameterization redundancies of such networks that will be key to enforcing G-invariance. All notation introduced here will persist throughout the paper.

Let $f : \mathbb{R}^m \mapsto \mathbb{R}$ be a *deep neural network* (DNN) of depth d constructed as follows: Let $\{n_1, \dots, n_{d+1}\}$ be a set of $d+1$ positive integers where[5] $n_1 = m$ and $n_{d+1} = 1$, and let $N_i = n_1 + \cdots + n_i$ for every $i \in \{1, \dots, d\}$. Let $W^{(i)} \in \mathbb{R}^{n_{i+1} \times N_i}$ and $b^{(i)} \in \mathbb{R}^{n_{i+1}}$ for every $i \in \{1, \dots, d\}$. Define $f^{(1)} : \mathbb{R}^m \mapsto \mathbb{R}^{n_1}$ by $f^{(1)}(x) = x$. For every $i \in \{1, \dots, d-1\}$, define $f^{(i+1)} : \mathbb{R}^m \mapsto \mathbb{R}^{N_{i+1}}$ by

$$f^{(i+1)}(x) = \begin{bmatrix} \mathrm{ReLU}(W^{(i)} f^{(i)}(x) + b^{(i)}) \\ f^{(i)}(x) \end{bmatrix}. \tag{3}$$

Then, let $f(x) = W^{(d)} f^{(d)}(x) + b^{(d)}$.

We thus define a DNN to be a feedforward ReLU network having all possible *skip connections*. We call the $W^{(i)}$ *weight matrices*, their rows *weight vectors*, and the $b^{(i)}$ *bias vectors*. The numbers $n_2, \dots, n_d$ are the widths of the hidden layers, and the numbers $N_2, \dots, N_d$ are the cumulative widths accounting for the skip connections. Note the traditional definition of a DNN can be recovered by setting all skip connections to zero.

Our first proposition below establishes a set of parameterization redundancies (i.e., reparameterizations of the DNN leaving the input-output function invariant) enjoyed by the above DNN architecture. For every positive integer n, let $\mathrm{C}(n)$ be the group of $n \times n$ diagonal matrices with positive diagonal entries.

Proposition 3 *Let $i \in \{1, \dots, d-1\}$, $C \in \mathrm{C}(n_{i+1})$, $P \in \mathrm{P}(n_{i+1})$, and $Z \in \mathrm{Z}(n_{i+1})$. Then the DNN f is invariant under the transformation*

$$\begin{aligned} W^{(i)} &\to CPZW^{(i)} \\ b^{(i)} &\to CPZb^{(i)} \\ W^{(i+1)} &\to W^{(i+1)} \begin{bmatrix} (CP)^{-1} & \mathcal{H}(-Z)W^{(i)} \\ 0 & I_{N_i} \end{bmatrix} \\ b^{(i+1)} &\to b^{(i+1)} + W^{(i+1)} \begin{bmatrix} \mathcal{H}(-Z)b^{(i)} \\ 0 \end{bmatrix}. \end{aligned}$$

Proposition 3 is the key to understanding how signed perm-reps can be used in G-DNNs. Specificly, if we want the parameters $(W^{(i)}, b^{(i)})$ of the ith layer to transform by a signed perm-rep, then the parameters $(W^{(i+1)}, b^{(i+1)})$ of the subsequent layer must transform as in Prop. 3 to compensate and maintain invariance. This idea is the basis of the next section.

5. Note the input dimension $n_1 = m$ is the same number as the degree of the matrix group G in Sec. 2.

3.2 G-invariant architectures

The following lemma gives a sufficient condition for each $f^{(i)}$ (the subnetwork comprising the first $i-1$ layers of f and acting as input of the ith layer) to be G-equivariant; the sufficient condition takes the form of equivariant constraints on the network parameters.

Lemma 4 *Let $\{\rho^{(1)}, \dots, \rho^{(d)}\}$ be a set of signed perm-reps $\rho^{(i)} : G \mapsto \mathrm{PZ}(n_{i+1})$ with $\rho^{(d)}$ the trivial rep. For each $i \in \{1, \dots, d\}$, let $\pi^{(i)} : G \mapsto \mathrm{P}(n_{i+1})$ and $\zeta^{(i)} : G \mapsto \mathrm{Z}(n_{i+1})$ be the unique functions such that $\rho^{(i)}(g) = \pi^{(i)}(g)\zeta^{(i)}(g) \forall g \in G$. Let $\{\psi^{(1)}, \dots, \psi^{(d)}\}$ be a set of reps $\psi^{(i)} : G \mapsto \mathrm{GL}(N_i)$ defined as:*

$$\psi^{(1)}(g) = g$$
$$\psi^{(i+1)}(g) = \begin{bmatrix} \pi^{(i)}(g) & \frac{1}{2}(W^{(i)}\psi^{(i)}(g) - \pi^{(i)}(g)W^{(i)}) \\ 0 & \psi^{(i)}(g) \end{bmatrix} \forall i \in \{1, \dots, d-1\}.$$

Suppose $\rho^{(i)}(g)W^{(i)} = W^{(i)}\psi^{(i)}(g)$ and $\rho^{(i)}(g)b^{(i)} = b^{(i)}$ for all $g \in G$ and $i \in \{1, \dots, d\}$. Then $f^{(i)}(gx) = \psi^{(i)}(g)f^{(i)}(x)$ for all $g \in G$, $x \in \mathbb{R}^m$, and $i \in \{1, \dots, d\}$.

Since $f^{(d)}(gx) = \psi^{(d)}f^{(d)}(x)$ and $W^{(d)}\psi^{(d)}(g) = W^{(d)}$ for all $g \in G$ and $x \in \mathbb{R}^m$, then we see that Lemma 4 gives a sufficient condition for f to be a *G-invariant deep neural network* (G-DNN). Note the sequence of signed perm-reps $\{\rho^{(1)}, \dots, \rho^{(d)}\}$ completely determines the G-DNN architecture (i.e., depth, number of hidden neurons per layer, and weightsharing patterns), and hence we will also refer to such sequences of reps as *G-DNN architectures*. Observe that the rep $\psi^{(i)}$ depends on the weight matrix $W^{(i-1)}$, and hence the equivariance condition on $W^{(i)}$ introduces a coupling between $W^{(i-1)}$ and $W^{(i)}$. This coupling makes it difficult to implement the equivariance constraints on the weight matrices directly, and we thus proceed to find a reparameterization admitting uncoupled equivariant weight matrices. To do this, we first state another lemma below, which gives an explicit formula for the reps $\psi^{(i)}$ (as opposed to a recursive formula) and shows each $\psi^{(i)}$ to be equivalent to a direct sum of layerwise reps.

For every $i \in \{1, \dots, d\}$, decompose $W^{(i)}$ into blocks as

$$W^{(i)} = \begin{bmatrix} W_i^{(i)} & W_{i-1}^{(i)} & \cdots & W_1^{(i)} \end{bmatrix},$$

where $W_j^{(i)} \in \mathbb{R}^{n_{i+1} \times n_j}$. Define the block matrix

$$A^{(i)} = \begin{bmatrix} I_{n_{i+1}} & -\frac{1}{2}W_i^{(i)} & -\frac{1}{2}W_{i-1}^{(i)} & \cdots & -\frac{1}{2}W_1^{(i)} \\ 0 & I_{n_i} & -\frac{1}{2}W_{i-1}^{(i-1)} & \cdots & -\frac{1}{2}W_1^{(i-1)} \\ 0 & 0 & I_{n_{i-1}} & \cdots & -\frac{1}{2}W_1^{(i-2)} \\ 0 & 0 & 0 & \ddots & \vdots \\ 0 & 0 & 0 & 0 & I_{n_1} \end{bmatrix}.$$

Define the rep $\Pi^{(i)} : G \mapsto \mathrm{O}(N_{i+1})$ by

$$\Pi^{(i)}(g) = \mathrm{diag}(\pi^{(i)}(g), \pi^{(i-1)}(g), \dots, \pi^{(1)}(g), g).$$

Lemma 5 *For every $i \in \{1, \dots, d-1\}$, we have*[6]

$$\psi^{(i+1)}(g) = A^{(i)-1}\Pi^{(i)}(g)A^{(i)} \forall g \in G.$$

Let $\pi^{(0)} : G \mapsto \mathrm{P}(n_1)$ be the identity rep $\pi^{(0)}(g) = g$, and let $A^{(0)} = I_{n_1}$. Then Lemma 5 holds for $i = 0$ as well.

We are now ready to state this section's key theorem, which gives a reparameterization of a G-DNN in which it admits uncoupled equivariant weight matrices.

Theorem 6 *Let f be G-invariant, with its parameters $\{(W^{(1)}, b^{(1)}), \dots, (W^{(d)}, b^{(d)})\}$ satisfying the conditions of Lemma 4 with respect to the signed perm-reps $\{\rho^{(1)}, \dots, \rho^{(d)}\}$.*

(a) For every $i \in \{1, \dots, d\}$, there exists $V^{(i)} \in \mathbb{R}^{n_{i+1} \times N_i}$ with block structure

$$V^{(i)} = \begin{bmatrix} V_i^{(i)} & V_{i-1}^{(i)} & \cdots & V_1^{(i)} \end{bmatrix}$$

where $V_j^{(i)} \in \mathbb{R}^{n_{i+1} \times n_j}$ such that

$$\rho^{(i)}(g)V_j^{(i)} = V_j^{(i)}\pi^{(j-1)}(g) \forall g \in G$$
$$W^{(i)} = V^{(i)}A^{(i-1)}.$$

(b) For every $i \in \{1, \dots, d\}$, define $g^{(i)} : \mathbb{R}^m \mapsto \mathbb{R}^{n_{i+1}}$ by $g^{(i)}(x) = W^{(i)}f^{(i)}(x)$ and $h^{(i)} : \mathbb{R}^m \mapsto \mathbb{R}^{N_i}$ by $h^{(i)}(x) = A^{(i-1)}f^{(i)}(x)$. Then we have the recursion

$$g^{(i)}(x) = V^{(i)}h^{(i)}(x) \forall i \in \{1, \dots, d\}$$
$$h^{(i+1)}(x) = \begin{bmatrix} \mathrm{ReLU}(g^{(i)}(x) + b^{(i)}) - \frac{1}{2}g^{(i)}(x) \\ h^{(i)}(x) \end{bmatrix} \forall i \in \{1, \dots, d-1\}.$$

The $V^{(i)}$ are the latent weight matrices, and in Thm. 6 (a) we see that each $V^{(i)}$ satisfies a linear equivariant condition that is easy to implement. The apparent weight matrices $W^{(i)}$ can then be reconstructed as $W^{(i)} = V^{(i)}A^{(i-1)}$. In practice, however, there is no need to perform this reconstruction; instead, we implement the recursion in Thm. 6 (b) as the forward pass of the G-DNN, where the final network output is $f(x) = g^{(d)}(x) + b^{(d)}$. Observe that the recursion is given directly in terms of the $V^{(i)}$ as opposed to the $W^{(i)}$. This recursion leverages the block-triangular structure of $A^{(i-1)}$ in the transformation $W^{(i)} = V^{(i)}A^{(i-1)}$. We describe a concrete implementation of the equivariance condition and the forward pass of the G-DNN in App. B.2.1.

3.3 Admissible architectures

Theorem 6 tells us how to construct a G-DNN, but it first requires us to select a sequence of reps or "architecture" $\rho^{(1)}, \dots, \rho^{(d)}$. Selecting the optimal sequence is the problem of G-invariant neural architecture design, which is beyond the scope of this work. At the very least, however, we would like to avoid sequences that correspond to "degenerate" network

6. The notation $A^{(i)-1}$ is shorthand for $(A^{(i)})^{-1}$, the matrix inverse of $A^{(i)}$.

architectures. In particular, we require that (1) every row of each weight matrix $W^{(i)}$ be nonzero and (2) no two rows of the augmented weight matrix $[W^{(i)} \mid b^{(i)}]$ be parallel. The first condition ensures there are no neurons disconnected from all previous layers, and the second condition ensures no two hidden neurons in a given layer can be combined into a single hidden neuron or skip connection. We call a sequence of reps $\rho^{(1)}, \dots, \rho^{(d)}$ an *admissible architecture* if it admits a G-DNN with weight matrices satisfying these two conditions. Below, Thm. 7 provides a characterization of admissible architectures that we implement in practice. First, however, we introduce additional notions and notation.

For every $A \in \mathbb{R}^{m \times m}$, let $\mathrm{st}_G(A)$ be the stabilizer subgroup

$$\mathrm{st}_G(A) = \{g \in G : gA = A\},$$

and for every finite orthogonal matrix group Γ, let P_Γ be the orthogonal projection operator onto the vector subspace pointwise-invariant under the action of Γ:

$$P_\Gamma = \frac{1}{|\Gamma|} \sum_{g \in \Gamma} g.$$

Let $\mathcal{S}(G)$ denote the set of all subgroups of G. Then, we define the function $\theta : \{(H, K, J) \in \mathcal{S}(G)^3 : |H : K| \leq 2\} \mapsto \mathcal{S}(G)$ by

$$\begin{aligned}\theta(H, K, J) &= \pi_J^{-1}[\mathrm{st}_{\pi_J(G)}(P_{\pi_J(K)} - (|H : K| - 1)P_{\pi_J(H)})] \\ &= \{g \in G : \pi_J(g)(P_{\pi_J(K)} - (|H : K| - 1)P_{\pi_J(H)}) = P_{\pi_J(K)} - (|H : K| - 1)P_{\pi_J(H)}\},\end{aligned}$$

where the ordinary perm-rep $\pi_J : G \mapsto \mathrm{P}(|G/J|)$ is defined as usual– i.e., defined to be equivalent to the action of G on G/J. Note π_J is defined only up to conjugation by a perm-matrix, since no ordering on the cosets in G/J is specified. It turns out, however, that θ is invariant under conjugation of π_J, and more generally θ is invariant and equivariant with respect to certain conjugations of the input subgroups (see Prop. 8 in App. B.3).

Let $f : \mathbb{R}^m \mapsto \mathbb{R}$ be a G-DNN with sequence of signed perm-reps $\rho^{(1)}, \dots, \rho^{(d)}$. For every $i \in \{1, \dots, d\}$, the signed perm-rep $\rho^{(i)}$ admits the following decomposition into irreducibles:

$$\rho^{(i)} = \bigoplus_{j=1}^{r^{(i)}} k_j^{(i)} \rho_j^{(i)}, \quad \rho_j^{(i)} = \rho^{(i)}_{H_j^{(i)} K_j^{(i)}}.$$

The irreps $\rho_1^{(i)}, \dots, \rho_{r^{(i)}}^{(i)}$ are inequivalent, and each $k_j^{(i)}$ is a positive integer where we define the notation

$$k_j^{(i)} \rho_j^{(i)} = \bigoplus_{k=1}^{k_j^{(i)}} \rho_j^{(i)}.$$

We say the ith layer of the G-DNN f has $r^{(i)}$ distinct *irreps* and $k_1^{(i)} + \cdots + k_{r^{(i)}}^{(i)}$ *channels*. Now for every $i \in \{1, \dots, d\}$, we define the functions $\phi^{(i)} : \{(H, K) \in \mathcal{S}(G)^2 : |H : K| \leq 2\} \mapsto \mathcal{S}(G)$ recursively by

$$\begin{aligned}\phi^{(1)}(H, K) &= \mathrm{st}_G(P_K - (|H : K| - 1)P_H) \\ \phi^{(i+1)}(H, K) &= \phi^{(i)}(H, K) \cap \bigcap_{j=1}^{r^{(i)}} \theta(H, K, H_j^{(i)}) \forall i \in \{1, \dots, d-1\}.\end{aligned}$$

Following from the equivariance of θ (Prop. 8) and that inner automorphisms respect subgroup intersection, we have the important property that each $\phi^{(i)}$ is equivariant with respect to pairwise conjugation:

$$\phi^{(i)}(gHg^{-1}, gKg^{-1}) = g\phi^{(i)}(H,K)g^{-1} \forall g \in G.$$

We are ready to state the theorem characterizing single-channel admissible architectures; note the following is a generalization of Thm. 4a of Agrawal and Ostrowski (2022).

Theorem 7 *Maintain the notation introduced in the last paragraph, and suppose* $k_j^{(i)} = 1 \forall i, j$ *(single-channel architecture). Then the architecture*[7] $\{\rho^{(1)}, \ldots, \rho^{(d)}\}$ *is admissible iff the following conditions hold:*

1. $\phi^{(i)}(H_j^{(i)}, K_j^{(i)}) = K_j^{(i)} \forall i, j.$
2. *If* $H_j^{(1)} = G$ *for some* j*, then* $P_G \neq 0$.[8]

Theorem 7 gives us a practical way to build admissible G-DNN architectures layer-by-layer as follows: Suppose we have already built the first i layers– i.e., we have selected the signed perm-reps $\rho^{(1)}, \ldots, \rho^{(i)}$ satisfying Thm. 7. Then the function $\phi^{(i+1)}$ is defined. We enumerate all signed perm-irreps ρ_{HK}, up to equivalence,[9] such that $\phi(H,K) = K$, and we then select a subset to comprise $\rho^{(i+1)}$. In terms of implementation, the computation of $\phi^{(i)}(H,K)$ boils down to the computation of $\theta(H, K, H_j^{(i-1)})$ for each j, which can be accomplished using Alg. 2 (see App. B.3 for details). We should think of the outputs $\theta(H,K,J)$ as entries of a precomputed table with a row for each (H,K) (up to pairwise conjugation) and a column for each J (up to conjugation). We give additional implementation details in the next section, including how we accomodate multiple input and output channels per layer.

To gain some intuition about Thm. 7, we briefly consider its application to G-SNN (i.e., depth $d = 2$) architectures, which are completely characterized in terms of the single signed perm-rep by which the weight matrix in the first layer transforms. In this case, Thm. 7 reduces to Thm. 4 (a) of Agrawal and Ostrowski (2022); a signed perm-rep ρ_{HK} is admissible for a G-SNN iff $\phi^{(1)}(H,K) = K$, or more explicitly,

$$\mathrm{st}_G(P_K - (|H:K| - 1)P_H) = K.$$

Here the weight vectors of the G-SNN are $\{gw : gH \in G/H\}$, where w is fixed under K and flips sign under $H \setminus K$. The weight vectors are pairwise nonparallel if $\mathrm{st}_G(w) = K$; if, however, $\mathrm{st}_G(w) > K$, then the G-orbit of w is smaller than the number of weight vectors, forcing some weight vectors to be equal up to sign. Theorem 7 serves to eliminate such degenerate architectures.

7. By definition of G-DNN architecture, $\rho^{(d)}$ is trivial; see Lemma 4 and the discussion immediately proceeding it.
8. This condition is trivially satisfied if G is a permutation matrix group.
9. The equivariance of ϕ is why enumeration of the signed perm-irreps only up to equivalence is sufficient.

Table 1: Ratio of the number of admissible G-DNN architectures to the total number of architectures for every depth and every group G, $|G| = 8$, up to isomorphism. Only architectures corresponding to sequences of irreps of strictly decreasing degree are considered.

Depth	C_8	$C_2 \times C_4$	C_2^3	D_4	Q_8
2	5/5	8/15	11/43	14/21	9/9
3	8/8	30/62	93/434	65/104	20/20
4	4/4	48/48	392/392	84/84	12/12

The number of admissible architectures can be significantly less than the total number of architectures. For example, we consider one particular permutation representation of every group of order 8 up to isomorphism, following the constructions described in App. C.1 of Agrawal and Ostrowski (2022). For each group, we only count the architectures corresponding to sequences of irreps of strictly decreasing degree, and we report the fraction of these that are admissible (Table 1). Observe that the reduction from the total number of architectures to only the admissible ones is often significant, which could be exploited perhaps in a future implementation of G-NAS. Note also that all architectures of maximum depth are admissible; we speculate this is because as depth increases and each new weight matrix $W^{(i)}$ grows in width (due to more skip connections to previous layers), the conditions under which two weight vectors are parallel become harder to satisfy. A complete answer, however, is left for future investigation.

3.4 Additional remarks

Channels The G-DNN supports multiple channels in a way generalizing the notion from traditional convolutional neural networks. As already mentioned in Sec. 3.3, the ith layer of the G-DNN f is said to have $k_j^{(i)}$ output channels or copies of the irrep $\rho_j^{(i)}$ and $k_j^{(i-1)}$ input channels from the irrep $\rho_j^{(i-1)}$. As a simpler case, suppose $\rho^{(i)}$ contains the same number $k^{(i)}$ of copies of each of its constituant irreps, and suppose the input x has $k^{(0)}$ channels. Then the ith layer can be said to have $k^{(i)}$ output channels and $k^{(i-1)}$ input channels. In practice, we implement the G-DNN assuming only one copy of each irrep $\rho_j^{(i)}$, and then we regard each element of the input x as a $k^{(0)}$-dimensional vector and each element of the latent weight matrix block $V_j^{(i)}$ (see Thm. 6 (a) for notation) as a $k^{(i)} \times k^{(j-1)}$ matrix.

Batch normalization It is possible to apply batch normalization (batchnorm) immediately after ReLU in a G-DNN without breaking G-invariance. That batchnorm is compatible with G-DNNs, and in particular type 2 signed perm-reps, out-of-the-box is not obvious, and we verify the compatibility in Prop. 10 (see App. B.4). The proof relies on the facts that (1) the first n_i columns of $W^{(i)}$ and $V^{(i)}$ are equal and (2) the first n_i columns of $V^{(i)}$ sum to zero if $\rho^{(i)}$ is a type 2 signed perm-irrep.

4. Examples

4.1 Concatenated ReLU

G-invariant architectures with signed perm-reps have previously been constructed without skip connections (Cohen and Welling, 2017) by replacing ReLU with the CReLU activation function defined in Eq. (2). The upshot is that for every signed perm-rep $\rho : G \mapsto \mathrm{PZ}(n)$, CReLU enjoys the equivariance property

$$\mathrm{CReLU}(\rho(g)x) = \pi_\rho(g)\,\mathrm{CReLU}(x) \forall g \in G, x \in \mathbb{R}^n,$$

where π_ρ is the unraveled ordinary perm-rep appearing in Thm. 2; CReLU architectures are thus compatible with signed perm-reps. The following example, however, establishes that CReLU architectures are strictly special cases of the G-DNN architectures presented in this paper.

Example 1 *Let $f : \mathbb{R}^m \mapsto \mathbb{R}$ be a DNN of the form*

$$f(x) = U^{(d)}\,\mathrm{CReLU}(U^{(d-1)} \cdots \mathrm{CReLU}(U^{(1)}x) \cdots),$$

where $U^{(1)} \in \mathbb{R}^{n_2 \times n_1}$ and $U^{(i)} \in \mathbb{R}^{n_{i+1} \times 2n_i} \forall i \in \{2, \ldots, d\}$ are equivariant weight matrices satisfying

$$\rho^{(1)}(g)U^{(1)} = U^{(1)}\pi^{(0)}(g) \forall g \in G$$
$$\rho^{(i)}(g)U^{(i)} = U^{(i)}\pi_{\rho^{(i-1)}}(g) \forall g \in G, i \in \{2, \ldots, d\},$$

where $\pi_{\rho^{(i-1)}}$ is the unraveling of $\rho^{(i-1)}$ as defined in Thm. 2. Then f admits an expression as a G-DNN of depth d whose latent weight matrices are given by the recursion

$$V^{(1)} = U^{(1)}$$
$$V^{(i+1)} = \begin{bmatrix} U_1^{(i+1)} + U_2^{(i+1)} & \frac{1}{2}(U_1^{(i+1)} - U_2^{(i+1)})V^{(i)} \end{bmatrix} \forall i \in \{1, \ldots, d-1\}.$$

Although Ex. 1 holds for any sequence of signed perm-reps $\rho^{(1)}, \ldots, \rho^{(d)}$ with $\rho^{(d)}$ trivial, the primary case of interest is when each $\rho^{(i)}$, $i \leq d-1$, is a direct sum only of type 2 signed perm-irreps; otherwise, if we had a subset of weight vectors that transformed by a type 1 signed perm-irrep, then we would apply ordinary RELU to their respective preactivations instead of CReLU. This is why it is also sufficient to consider CReLU networks without bias vectors, as type 2 signed perm-irreps constrain bias vectors to be zero.

Example 1 implies that the G-DNN architectures introduced in this paper are at least as powerful as (G-invariant) CReLU-based architectures, whose utility has already been demonstrated in the literature (Shang et al., 2016; Cohen and Welling, 2017). Moreover, we recover CReLU architectures only when the latent weight matrices $V^{(i)}$ take a very special form; this suggests that there may exist G-DNN architectures that cannot be constructed with CReLU alone. To investigate this possibility, we count the numbers of *admissible* CReLU architectures for various groups following the same setup used to obtain the results in Table 1. As before, we consider every sequence of signed perm-irreps

Table 2: Ratio of the number of admissible CReLU architectures to the total number of architectures for every depth and every group G, $|G| = 8$, up to isomorphism. Only architectures corresponding to sequences of irreps of strictly decreasing degree are considered.

Depth	C_8	$C_2 \times C_4$	C_2^3	D_4	Q_8
2	5/5	8/15	11/43	14/21	9/9
3	8/8	30/62	88/434	65/104	20/20
4	4/4	34/48	238/392	66/84	12/12

Table 3: Numbers of admissible CReLU and G-DNN architectures at every depth for the symmetry group G of the icosahedron. Only architectures corresponding to sequences of irreps of strictly decreasing degree are considered. We do not report the total number of such sequences as we found every sequence to correspond to an admissible G-DNN architecture.

Depth	CReLU	G-DNN
2	20	20
3	136	142
4	441	516
5	776	1089
6	769	1392
7	407	1064
8	90	448
9	0	80
Total	2639	4751

$\rho_{H^{(1)}K^{(1)}}, \ldots, \rho_{H^{(d)}K^{(d)}}$ of strictly decreasing degree terminating with the trivial rep. In contrast to G-DNNs, however, the criterion for a CReLU architecture to be admissible is much simpler; since there are no explicit skip connections, then the function $\phi^{(i+1)}$ in Thm. 7 no longer depends recursively on $\phi^{(i)}$. The resulting condition for a CReLU architecture with signed perm-irreps $\rho_{(H^{(1)}K^{(1)}}, \ldots, \rho_{H^{(d)}K^{(d)}}$ to be admissible is

$$\theta(H^{(i+1)}, K^{(i+1)}, H^{(i)}) = K^{(i+1)} \forall i \in \{1, \ldots, d-1\}.$$

Based on this, we report fractions of admissible CReLU architectures (Table 2) in direct analogy to Table 1. At depth 4, there are fewer admissible CReLU architectures than admissible G-DNN architectures.

As a more striking example, we consider the symmetry group of the icosahedron used later for 3D object classification in Sec. 4.3. We report the numbers of admissible CReLU and G-DNN architectures for this group (Table 3). There are about 1.8× as many admissible G-DNN architectures as there are admissible CRELU architectures, including nine-layer

G-DNNs with no CRELU architectures of the same depth. We conclude that while the CRELU activation function gives a simple way to implement equivariance with respect to signed perm-reps, our G-DNNs grant us access to large parts of G-invariant architecture space previously unreachable.

4.2 Binary multiplication

Our first example application is an exact result and demonstrates that G-DNNs with type 2 signed perm-reps occur even in the context of simple mathematical operations.

Example 2 *For $d \geq 3$, let $m = 2^d$ and*

$$\mathcal{X} = \{x \in \{0,1\}^m : x_{2i-1} + x_{2i} = 1 \forall i \in \{1, \dots, m/2\}\}.$$

For $\{e_1, \dots, e_m\}$ the standard orthonormal basis, Let $G < \mathrm{P}(m)$ be the subgroup of all even permutation matrices that only transpose e_{2i-1} and e_{2i}. Define the function $f : \mathcal{X} \mapsto \{-1, 1\}$ by

$$f(x) = \prod_{i=1}^{m/2} (x_{2i-1} - x_{2i}).$$

Then f admits an expression as a G-DNN of depth d where:

(a) For $i \in \{1, \dots, d\}$ and $j \in \{1, \dots, i\}$, the blocks $V_j^{(i)}$ of the latent weight matrices (see Thm. 6 (a)) are given by

$$V_j^{(i)} = \begin{cases} I_{m/2^{i+1}} \otimes \begin{bmatrix} 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}, & \text{if } 1 \leq i = j < d \\ \begin{bmatrix} 1 & -1 \end{bmatrix}, & \text{if } i = j = d \\ 0, & \text{otherwise.} \end{cases}$$

(b) For $i \in \{1, \dots, d-1\}$, the ith layer transforms by the signed perm-rep

$$\rho^{(i)} = \begin{cases} \bigoplus_{j=1}^{m/2^{i+1}} \rho_{H_j^{(i)} K_j^{(i)}}, & \text{if } i \leq d-2 \\ \rho_{H_1^{(d-1)} K_1^{(d-1)}} \oplus \rho_{H_1^{(d-1)} K_1^{(d-1)}}, & \text{if } i = d-1, \end{cases}$$

where

$$K_j^{(1)} = \{g \in G : g v_j = v_j\}$$
$$H_j^{(1)} = \{g \in G : g v_j = \pm v_j\},$$

where $v_j = e_{4j-3} - e_{4j-2} + e_{4j-1} - e_{4j}$, and

$$K_j^{(i+1)} = H_{2j-1}^{(i)} \cap H_{2j}^{(i)}$$
$$H_j^{(i+1)} = (H_{2j-1}^{(i)} \cap H_{2j}^{(i)}) \cup ((G \setminus H_{2j-1}^{(i)}) \cap (G \setminus H_{2j}^{(i)})),$$

for $i \in \{1, \dots, d-2\}$. The rep $\rho^{(d)}$ is trivial as usual.

Table 4: Binary cross-entropy losses (mean and standard deviation over 24 random initialization seeds) of four G-DNN architectures (see main text) on the binary multiplication problem. We report both the training loss and validation loss before training (initial) and after 5 epochs of training (final). Training and validation losses are equal because each of the two class labels corresponds to exactly one G-orbit, over which each G-DNN is constant– regardless of the dataset split. All values correspond to a classification accuracy of 50%, except the final losses of zero for the type 2 architecture corresponding to 100% accuracy.

Architecture	Initial train	Initial val	Final train	Final val
Type 1	5.11± 4.68	5.11± 4.68	0.71± 0.04	0.71± 0.04
Type 2	1.33± 0.60	1.33± 0.60	0.00	0.00
Unraveled (random init)	11.36±12.34	11.36±12.34	0.71± 0.03	0.71± 0.03
Unraveled (type 2 init)	1.33± 0.60	1.33± 0.60	0.70	0.70

Here, $f(x)$ is the product of $\frac{m}{2}$ binary elements in $\{-1, 1\}$ but where the elements are each represented as 2D one-hot vectors and are then concatenated into the m-dimensional input x. From the perspective of multiplication of elements in $\{-1, 1\}$, the group G corresponds to an even number of sign flips, which clearly leaves the final product invariant. Example 2 gives an explicit construction of a G-DNN that implements the function f. Each layer of the G-DNN partitions its input into pairs and takes each of their products; the network thus iteratively coarsegrains the input until the final scalar product is returned.[10] Each latent weight matrix of the G-DNN has a block-diagonal structure and no latent skip connections; however, by Thm. 6 (a), the G-DNN does have skip connections in terms of the apparent weight matrices. All signed perm-irreps in the network–except $\rho^{(d)}$ which must be trivial–are type 2, and the image of each one is isomorphic to the Klein-4 group. The image of the penultimate rep $\rho^{(d-1)}$ is also isomorphic to the Klein-4 group, but the rep itself is not irreducible and decomposes into two copies of a type 2 scalar irrep.

We investigate Ex. 2 empirically by generating the complete dataset $\{(x, f(x)) : x \in \mathcal{X}\}$ for $m = 16$. Since $f(x) = \pm 1$, then we regard the estimation of f as a binary classification problem. We use a random 20% of the dataset (with class stratification) for training and the rest for validation. We instantiate the "type 2" architecture with the type 2 signed perm-irreps in Ex. 2 (b) and randomly initialize its weights. We compare the type 2 architecture to two type 1 baselines: (1) the "type 1" architecture obtained by sending[11] $\rho_{HK} \to \rho_{HH}$ (see Thm. 1) and (2) the "unraveled" architecture obtained by sending $\rho_{HK} \to \rho_{KK}$ (seeThm. 2 (d)). The type 1, type 2, and unraveled architectures are thus

10. Although f is a simple function, we were unable to construct a G-DNN—in particular, a shallower G-invariant architecture that (1) has G-equivariant preactivations and (2) does not have an exponential number of hidden neurons per layer—simpler than the one given in Ex. 2 that fits f. For example, one could compute $f(x)$ with a shallow network that first maps x linearly into $\{-1, 1\}^{\frac{m}{2}}$, then takes the sum $\vec{1}^\top x$, and finally returns the parity of the sum using a suitable combination of ReLU neurons; however, the function $x \to \vec{1}^\top x$ is not G-equivariant.
11. This transformation is called topological tunneling in the literature (Agrawal and Ostrowski, 2022)

analogous to the three weightsharing patterns in Fig. 1 (see Sec. 2.2); i.e., the type 1 architecture has the same number of hidden neurons as the type 2 architecture but expresses different functions, and the unraveled architecture has double the number of hidden neurons and the capacity to express both the smaller two architectures as well as much more.

We trained all architectures for 5 epochs with the Adam optimizer with minibatch size 64, learning rate 0.01, and learning rate decay 0.99 per step. Starting at 50% classification accuracy, the type 2 architecture quickly achieves 100% training and validation accuracy as well as a final binary cross-entropy loss of 0.00 (Table 4). All other architectures remain stuck at 50% accuracy even after training. To explain these results, we first note that in the type 2 architecture, the latent weight matrices for $i \in \{1, \ldots, d-1\}$ are constrained to have the form

$$V_j^{(i)} = \begin{cases} I_{m/2^{i+1}} \otimes \begin{bmatrix} u_j^{(i)} & -u_j^{(i)} & v_j^{(i)} & -v_j^{(i)} \\ u_j^{(i)} & -u_j^{(i)} & -v_j^{(i)} & v_j^{(i)} \end{bmatrix}, & \text{if } 1 \leq j = i < d \\ 0, & \text{if } 1 \leq j < i < d, \end{cases}$$

where the $u_j^{(i)}$ and $v_j^{(i)}$ are free learnable parameters. In the type 1 architecture, the negative signs in front of $v_j^{(i)}$ are omitted, and the off-diagonal blocks are no longer zero but are instead constrained in the same way as the diagonal blocks. As a result, the very first linear transformation $g^{(1)}(x) = V^{(1)}x$ in the architecture is invariant to every transposition $(x_{2i-1}, x_{2i}) \to (x_{2i}, x_{2i-1})$– not just even numbers of them; the rest of the architecture is thus unable to distinguish the two classes of inputs, resulting in a flat 50% accuracy. The type 1 architecture simply does not have the capacity to solve the binary classification problem.

In contrast to the type1 and type 2 architectures, the latent weight matrices $V_i^{(i)}$ of the unraveled architectures are not even constrained to be block-diagonal, and the $V_j^{(i)}$ for $j < i$ are not constrained to be zero. The unraveled architecture thus has about $6.5\times$ the number of free learnable parameters compared to the type 2 architecture, and it must learn the block-diagonal weight structure from data alone. Based on the training loss, the complete failure of the unraveled architecture is likely due to a severe trainability issue. To confirm the unraveled architecture indeed has the capacity to express the solution and in particular the type 2 architecture, we loaded the trained weights of the type 2 architecture into the unraveled architecture by an appropriate mapping and confirmed that the resulting architecture indeed achieves 100% training and validation accuracy. To test if the failure to train is due to poor initialization, we implemented the unraveled architecture with both random initialization (called "random init" in Table 4) and with the randomly initialized weights of the type 2 architecture (called "type 2 init" in Table 4). While initialization with the initial weights of the type 2 architecture reduces the initial loss of the unraveled architecture to match the type 2 architecture, it does not solve the trainability issue. We thus conclude the unraveled architecture simply does not have sufficient inductive bias to solve the binary classification problem.

4.3 3D object classification

Our second example application demonstrates that G-DNNs with type 2 signed perm-reps carry inductive bias that can be useful "in the wild". We consider the ModelNet40 dataset(Wu et al., 2015), which contains 9843 training and 2468 validation samples of 3D CAD mesh representations of 40 different objects ranging from airplanes to toilets. The problem is to predict the object class given an input mesh. We preprocess the data in identical fashion to Jiang et al. (2019).[12] Specifically, we first bound each mesh in the unit sphere and discretize the sphere into an icosahedron. To increase resolution, we include the midpoint on each icosahedral edge as a vertex, normalize all vertices to have unit norm, and then repeat once more on the new polyhedron; this yields 162 vertices in all. From each of these vertices, we perform ray-tracing to the origin; we record the distance from the sphere to the mesh as well as the sine and cosine of the incident angle. The representation is further augmented with the three channels corresponding to the convex hull of the input mesh, yielding a 6-channel input representation over 162 points.

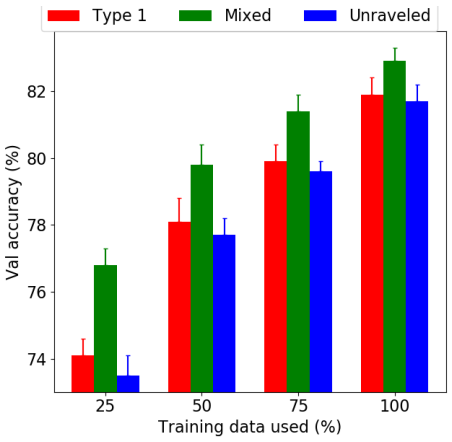

Figure 2: Validation classification accuracies (mean and standard deviation over 24 random initialization seeds) for three G-DNN architectures (see main text) on the ModelNet40 dataset using different percentages of the full training data. The mixed architecture—the only one including type 2 signed perm-irreps—clearly outperforms the two baselines, with the performance gap increasing as less training data is available.

The relevant symmetry group G is the symmetry group of the icosahedron, generated by (1) the order-3 rotation about the normal vector to one of the faces and (2) the order-5 rotation about one of the vertices; the group has 60 elements and is isomorphic to the alternating group A_5. We consider G-DNN architectures with four layers and 16, 32, 64, and 40 output channels per rep in each respective layer. We specify apriori the degrees of the signed perm-irreps to be used in each layer as 30, 15, 10, and 1; we then enumerate the admissible architectures as described after Thm. 7, and we select the following designs: For the "mixed" architecture, we select one type 1 and one type 2 signed perm-irrep in each of the first three layers[13]we select the trivial rep in the last layer, as always. We compare the mixed architecture to the corresponding "type 1" and "unraveled" architectures, which are constructed exactly as done in Sec. 4.2.

12. Jiang et al. (2019) perform five iterations of refinement to increase resolution, whereas we only perform two. This is because our current G-DNN implementation is based on fully-connected linear layers and thus does not scale well to high resolution. We plan to remedy this in future work by introducing local kernel windows. We attribute the difference in accuracy reported in Fig. 2 and that reported by Jiang et al. (2019) to this difference in resolution.
13. In each layer, the selected type 1 and type 2 signed perm-irreps are cohomologous as defined in Agrawal and Ostrowski (2022); i.e., they have the forms ρ_{HH} and ρ_{HK} respectively.

We trained all architectures for 500 epochs with the Adam optimizer with minibatch size 64, learning rate 0.01, and learning rate decay 0.99 per step. We included batchnorm as described in Sec. 3.4 after each ReLU layer. In each run, we performed retroactive early stopping by recording the highest validation accuracy achieved over all epochs. We find that the mixed architecture—the only one containing type 2 signed perm-irreps—significantly outperforms the two baseline architectures in terms of validation accuracy, with the performance gap increasing as less training data is used. This suggests that the mixed architecture carries stronger inductive bias, still consistent with the ground truth, as compared to the baselines.

5. Conclusion

We have introduced the G-DNN, a G-invariant densely connected DNN architecture. In contrast to previous G-invariant architectures in the literature such as the G-CNN (Cohen and Welling, 2016), the G-DNN is built with *signed* perm-reps that do not require individual layers of the network to be G-equivariant. The result is a richer family of G-invariant architectures never seen before (carefully quantified in terms of admissible architecture count), and we have demonstrated with both theoretical and empirical examples that some of these novel architectures can boost predictive performance.

To be clear, we do not claim the G-DNN to be a new state-of-the-art (SOTA) for G-invariant deep learning. Rather, our work is a demonstration that signed perm-reps, combined with skip connections, are mathematically natural building blocks for deep G-invariant architectures with practical potential. Indeed, we suspect that the structures and ideas presented in this paper, once extended and combined with domain-specific bells and whistles, could in fact help to boost the performance of SOTA G-invariant architectures.

Even at the domain-agnostic level, however, several open questions remain for future research. First, can we extend G-DNNs from G-invariance to G-equivariance? The only obstacle here is the construction of architectures guaranteed to be admissible. Second, can we perform G-NAS to find the optimal signed perm-irreps to use in each layer? Third and finally, are there ways of enforcing G-invariance that go even beyond the G-DNN architectures described in this paper? A complete classification of *all* G-invariant architectures would give us the finest possible control on inductive bias, thereby allowing us—in principle—to optimize predictive performance on G-invariant problems.

Acknowledgments

D.A. was supported by NSF award No. 2202990. J.O. was supported by DOE grant DE-SC0018175.

Appendix A. Signed permutation representations

A.1 Central hypothesis

Proof [Proof of Thm. 1] For every $J \leq G$, let P_J be the $m \times m$ orthogonal projection operator onto the subspace of $\mathbb{R}^m$ pointwise-invariant under the action of J:

$$P_J = \frac{1}{|J|} \sum_{g \in J} g.$$

By Thm. 4b of Agrawal and Ostrowski (2022), there exists $w_2 \in \mathrm{ran}(P_{K_2} - (|H : K_2| - 1)P_H)$ and a transversal $\{g_1, \ldots, g_n\}$ of G/H such that

$$W_2 = \begin{bmatrix} g_1 w_2 & g_2 w_2 & \cdots & g_n w_2 \end{bmatrix}^\top.$$

Moreover, there exists $w_1 \in \mathrm{ran}(P_{K_1} - (|H : K_1| - 1)P_H)$ and $P \in \mathrm{P}(n)$ such that

$$PW_1 = \begin{bmatrix} g_1 w_1 & g_2 w_1 & \cdots & g_n w_1 \end{bmatrix}^\top.$$

(The permutation matrix P is in general necessary so we can use the same transversal of G/H.) We have

$$\begin{aligned} \mathrm{diag}(PW_1 W_2^\top) &= \{(g_i w_1)^\top g_i w_2\}_{i=1}^n \\ &= \{w_1^\top g_i^\top g_i w_2\}_{i=1}^n \\ &= \{w_1^\top w_2\}_{i=1}^n \\ &= 0, \end{aligned}$$

where the last step follows by Prop. 6 of Agrawal and Ostrowski (2022). ■

Proof [Proof of Thm. 2] **(a)** Let $\pi : G \mapsto \mathrm{P}(n)$ and $\zeta : G \mapsto \mathrm{Z}(n)$ be the unique functions satisfying $\rho(g) = \pi(g)\zeta(g) \forall g \in G$ (note π should not be confused with π_ρ appearing in the theorem statement). Evaluating the Kronecker product, the function π_ρ can be rewritten as

$$\begin{aligned} \pi_\rho(g) &= \mathcal{H}\left(\begin{bmatrix} \rho(g) & -\rho(g) \\ -\rho(g) & \rho(g) \end{bmatrix}\right) \\ &= \mathcal{H}\left(\begin{bmatrix} \pi(g)\zeta(g) & -\pi(g)\zeta(g) \\ -\pi(g)\zeta(g) & \pi(g)\zeta(g) \end{bmatrix}\right) \\ &= \mathcal{H}\left(\begin{bmatrix} \pi(g) & 0 \\ 0 & \pi(g) \end{bmatrix}\begin{bmatrix} \zeta(g) & -\zeta(g) \\ -\zeta(g) & \zeta(g) \end{bmatrix}\right) \\ &= \begin{bmatrix} \pi(g) & 0 \\ 0 & \pi(g) \end{bmatrix} \mathcal{H}\left(\begin{bmatrix} \zeta(g) & -\zeta(g) \\ -\zeta(g) & \zeta(g) \end{bmatrix}\right), \end{aligned}$$

where in the last step we exploited the permutation-equivariance of elementwise operations. Now since $\mathcal{H}(z) = \frac{1+z}{2}$ for $z \in \{-1, 1\}$, then we have

$$\begin{aligned}
\pi_\rho(g) &= \begin{bmatrix} \pi(g) & 0 \\ 0 & \pi(g) \end{bmatrix} \frac{1}{2} \left(\begin{bmatrix} I_n & I_n \\ I_n & I_n \end{bmatrix} + \begin{bmatrix} \zeta(g) & -\zeta(g) \\ -\zeta(g) & \zeta(g) \end{bmatrix} \right) \\
&= \frac{1}{2} \left(\begin{bmatrix} \pi(g) & \pi(g) \\ \pi(g) & \pi(g) \end{bmatrix} + \begin{bmatrix} \pi(g)\zeta(g) & -\pi(g)\zeta(g) \\ -\pi(g)\zeta(g) & \pi(g)\zeta(g) \end{bmatrix} \right) \\
&= \frac{1}{2} \begin{bmatrix} \pi(g) + \rho(g) & \pi(g) - \rho(g) \\ \pi(g) - \rho(g) & \pi(g) + \rho(g) \end{bmatrix}. (*)
\end{aligned}$$

Now, let $\pi(g)_{ij}$ (resp. $\rho(g)_{ij}$) be the unique nonzero element in the ith row of $\pi(g)$ (resp. $\rho(g)$). Since $\rho(g)_{ij} = \pm\pi(g)_{ij}$, then exactly one of $\frac{1}{2}[\pi(g)_{ij} \pm \rho(g)_{ij}]$ is unity, while the other is zero. Thus, every row of (*) is a one-hot vector. The same argument can be made for the columns, and hence (*) is a bona fide permutation matrix; i.e., $\pi_\rho : G \mapsto \mathrm{P}(2n)$ is a well-defined function.

All that remains is to show π_ρ is a homomorphism. We rewrite (*) as

$$\pi_\rho(g) = \frac{1}{2} \left(\begin{bmatrix} I_n \\ I_n \end{bmatrix} \pi(g) \begin{bmatrix} I_n & I_n \end{bmatrix} + \begin{bmatrix} I_n \\ -I_n \end{bmatrix} \rho(g) \begin{bmatrix} I_n & -I_n \end{bmatrix} \right).$$

Then for $g, h \in G$, it is easy to verify that

$$\begin{aligned}
\pi_\rho(g)\pi_\rho(h) &= \frac{1}{4} \left(\begin{bmatrix} I_n \\ I_n \end{bmatrix} \pi(g) \begin{bmatrix} I_n & I_n \end{bmatrix} + \begin{bmatrix} I_n \\ -I_n \end{bmatrix} \rho(g) \begin{bmatrix} I_n & -I_n \end{bmatrix} \right) \left(\begin{bmatrix} I_n \\ I_n \end{bmatrix} \pi(h) \begin{bmatrix} I_n & I_n \end{bmatrix} \right. \\
&\quad \left. + \begin{bmatrix} I_n \\ -I_n \end{bmatrix} \rho(h) \begin{bmatrix} I_n & -I_n \end{bmatrix} \right) \\
&= \frac{1}{4} \left(2 \begin{bmatrix} I_n \\ I_n \end{bmatrix} \pi(g)\pi(h) \begin{bmatrix} I_n & I_n \end{bmatrix} + 0 + 0 + 2 \begin{bmatrix} I_n \\ -I_n \end{bmatrix} \rho(g)\rho(h) \begin{bmatrix} I_n & -I_n \end{bmatrix} \right) \\
&= \pi_\rho(gh).
\end{aligned}$$

(b) Using (*) from part (a), we have

$$\begin{aligned}
\pi_\rho(g) \begin{bmatrix} W \\ -W \end{bmatrix} &= \frac{1}{2} \begin{bmatrix} \pi(g) + \rho(g) & \pi(g) - \rho(g) \\ \pi(g) - \rho(g) & \pi(g) + \rho(g) \end{bmatrix} \begin{bmatrix} W \\ -W \end{bmatrix} \\
&= \frac{1}{2} \begin{bmatrix} \pi(g)W + \rho(g)W - \pi(g)W + \rho(g)W \\ \pi(g)W - \rho(g)W - \pi(g)W - \rho(g)W \end{bmatrix} \\
&= \frac{1}{2} \begin{bmatrix} 2\rho(g)W \\ -2\rho(g)W \end{bmatrix} \\
&= \rho(g) \begin{bmatrix} W \\ -W \end{bmatrix} \\
&= \begin{bmatrix} W \\ -W \end{bmatrix} g,
\end{aligned}$$

proving the claim.

(c) For the forward implication, we prove its contrapositive; suppose ρ is type 1. Then $\pi = \rho$ so that (*) implies

$$\pi_\rho = \frac{1}{2}\begin{bmatrix} 2\pi(g) & 0 \\ 0 & 2\pi(g) \end{bmatrix} = \begin{bmatrix} \pi(g) & 0 \\ 0 & \pi(g) \end{bmatrix},$$

which is clearly reducible.

For the reverse implication, suppose ρ is type 2. To show π_ρ is irreducible, we will show it is transitive on the standard orthonormal basis $\{\omega_1, \dots, \omega_{2n}\}$. Let $i, j \in \{1, \dots, 2n\}$, and without loss of generality suppose $i \leq n$.

Case 1: Suppose $j \leq n$. Then ω_i is just e_i (the ith standard orthonormal basis vector of dimension n) concatenated with $\vec{0}_n$, and similar for ω_j. Now since ρ is irreducible, then there exists $g \in G$ such that $\rho(g)e_i = e_j$; note $\pi(g)e_i = e_j$ as well. We thus have

$$\begin{aligned} \pi_\rho(g)\omega_i &= \frac{1}{2}\begin{bmatrix} \pi(g)+\rho(g) & \pi(g)-\rho(g) \\ \pi(g)-\rho(g) & \pi(g)+\rho(g) \end{bmatrix}\begin{bmatrix} e_i \\ \vec{0}_n \end{bmatrix} \\ &= \frac{1}{2}\begin{bmatrix} \pi(g)e_i + \rho(g)e_i \\ \pi(g)e_i - \rho(g)e_i \end{bmatrix} \\ &= \frac{1}{2}\begin{bmatrix} e_j + e_j \\ e_j - e_j \end{bmatrix} \\ &= \begin{bmatrix} e_j \\ \vec{0}_n \end{bmatrix} \\ &= \omega_j, \end{aligned}$$

establishing transitivity in this case.

Case 2: Suppose instead $j > n$. Then ω_j is the concatenation of $\vec{0}_n$ and e_{j-n}. Since ρ is irreducible, then there exists $g \in G$ such that $\rho(g)e_i = -e_{n-j}$. The rest of the proof proceeds in analogy to case 1.

(d) For every $v \in \mathbb{R}^n$ and linear rep $\tau : G \mapsto \mathrm{GL}(n, \mathbb{R})$, define the stabilizer subgroup

$$\mathrm{st}_\tau(v) = \{g \in G : \tau(g)v = v\}.$$

Since $\rho = \rho_{HK}$, then $\mathrm{st}_\rho(e_1) = K$. Since π_ρ is an ordinary perm-rep, then all we must show is $\mathrm{st}_{\pi_\rho}(\omega_1) = K$ to establish the claim. Similar to part (c), $\mathrm{st}_{\pi_\rho}(\omega_1)$ is the set of all $g \in G$ such that

$$\begin{aligned} \pi_\rho(g)\omega_1 &= \omega_1 \\ \frac{1}{2}\begin{bmatrix} \pi(g)+\rho(g) & \pi(g)-\rho(g) \\ \pi(g)-\rho(g) & \pi(g)+\rho(g) \end{bmatrix}\begin{bmatrix} e_1 \\ \vec{0}_n \end{bmatrix} &= \begin{bmatrix} e_1 \\ \vec{0}_n \end{bmatrix} \\ \frac{1}{2}\begin{bmatrix} \pi(g)e_1 + \rho(g)e_1 \\ \pi(g)e_1 - \rho(g)e_1 \end{bmatrix} &= \begin{bmatrix} e_1 \\ \vec{0}_n \end{bmatrix}. \end{aligned}$$

Taking the difference of the two rows, we obtain $\rho(g)e_1 = e_1$; hence, $\mathrm{st}_{\pi_\rho}(\omega_1) = \mathrm{st}_\rho(e_1) = K$, completing the proof. ■

Appendix B. G-invariant deep neural networks

B.1 Parameterization redundancies

To understand the inclusion of skip connections as a reparameterization, we rewrite Eq. 3 as

$$\begin{aligned} f^{(i+1)}(x) &= \begin{bmatrix} \mathrm{ReLU}(W^{(i)} f^{(i)}(x) + b^{(i)}) \\ \mathrm{ReLU}(f^{(i)}(x)) - \mathrm{ReLU}(-f^{(i)}(x)) \end{bmatrix} \\ &= \begin{bmatrix} I_{n_{i+1}} & 0 & 0 \\ 0 & I_{N_i} & -I_{N_i} \end{bmatrix} \mathrm{ReLU}\left(\begin{bmatrix} W^{(i)} \\ I_{N_i} \\ -I_{N_i} \end{bmatrix} f^{(i)}(x) + \begin{bmatrix} b^{(i)} \\ 0 \\ 0 \end{bmatrix} \right). \end{aligned}$$

The outer matrix in the last equation can be combined with the matrix in the next layer; the result is a DNN having the same depth as the original—and representing the same input-output function—but with no skip connections, as they have been transformed into additional ReLU neurons.

Proof [Proof of Prop. 3] We will show $W^{(i+1)} f^{(i+1)}(x) + b^{(i+1)}$ is invariant under the transformation. The function $f^{(i+1)}$ transforms as

$$\begin{aligned} f^{(i+1)}(x) &\to \begin{bmatrix} \mathrm{ReLU}(CPZW^{(i)} f^{(i)}(x) + CPZb^{(i)}) \\ f^{(i)}(x) \end{bmatrix} \\ &= \begin{bmatrix} CP & 0 \\ 0 & I_{N_i} \end{bmatrix} \begin{bmatrix} \mathrm{ReLU}(Z(W^{(i)} f^{(i)}(x) + b^{(i)}) \\ f^{(i)}(x) \end{bmatrix} \\ &= \begin{bmatrix} CP & 0 \\ 0 & I_{N_i} \end{bmatrix} \left(\begin{bmatrix} \mathrm{ReLU}(W^{(i)} f^{(i)}(x) + b^{(i)}) \\ f^{(i)}(x) \end{bmatrix} - \begin{bmatrix} \mathcal{H}(-Z)(W^{(i)} f^{(i)}(x) + b^{(i)}) \\ 0 \end{bmatrix} \right) \\ &= \begin{bmatrix} CP & 0 \\ 0 & I_{N_i} \end{bmatrix} \left(\begin{bmatrix} I_{n_{i+1}} & -\mathcal{H}(-Z)W^{(i)} \\ 0 & I_{N_i} \end{bmatrix} \begin{bmatrix} \mathrm{ReLU}(W^{(i)} f^{(i)}(x) + b^{(i)}) \\ f^{(i)}(x) \end{bmatrix} - \begin{bmatrix} \mathcal{H}(-Z) b^{(i)} \\ 0 \end{bmatrix} \right) \\ &= \begin{bmatrix} CP & 0 \\ 0 & I_{N_i} \end{bmatrix} \left(\begin{bmatrix} I_{n_{i+1}} & -\mathcal{H}(-Z)W^{(i)} \\ 0 & I_{N_i} \end{bmatrix} \begin{bmatrix} \mathrm{ReLU}(W^{(i)} f^{(i)}(x) + b^{(i)}) \\ f^{(i)}(x) \end{bmatrix} \right. \\ &\quad \left. - \begin{bmatrix} I_{n_{i+1}} & -\mathcal{H}(-Z)W^{(i)} \\ 0 & I_{N_i} \end{bmatrix} \begin{bmatrix} \mathcal{H}(-Z) b^{(i)} \\ 0 \end{bmatrix} \right) \\ &= \begin{bmatrix} CP & 0 \\ 0 & I_{N_i} \end{bmatrix} \begin{bmatrix} I_{n_{i+1}} & -\mathcal{H}(-Z)W^{(i)} \\ 0 & I_{N_i} \end{bmatrix} \left(\begin{bmatrix} \mathrm{ReLU}(W^{(i)} f^{(i)}(x) + b^{(i)}) \\ f^{(i)}(x) \end{bmatrix} - \begin{bmatrix} \mathcal{H}(-Z) b^{(i)} \\ 0 \end{bmatrix} \right) \\ &= \begin{bmatrix} CP & -CP\mathcal{H}(-Z)W^{(i)} \\ 0 & I_{N_i} \end{bmatrix} \left(f^{(i+1)}(x) - \begin{bmatrix} \mathcal{H}(-Z) b^{(i)} \\ 0 \end{bmatrix} \right). \end{aligned}$$

We thus have

$$\begin{aligned} W^{(i+1)} f^{(i+1)}(x) + b^{(i+1)} &\to W^{(i+1)} \begin{bmatrix} (CP)^{-1} & \mathcal{H}(-Z)W^{(i)} \\ 0 & I_{N_i} \end{bmatrix} \begin{bmatrix} CP & -CP\mathcal{H}(-Z)W^{(i)} \\ 0 & I_{N_i} \end{bmatrix} \left(f^{(i+1)}(x) \right. \\ &\quad \left. - \begin{bmatrix} \mathcal{H}(-Z) b^{(i)} \\ 0 \end{bmatrix} \right) + b^{(i+1)} + W^{(i+1)} \begin{bmatrix} \mathcal{H}(-Z) b^{(i)} \\ 0 \end{bmatrix} \\ &= W^{(i+1)} \left(f^{(i+1)}(x) - \begin{bmatrix} \mathcal{H}(-Z) b^{(i)} \\ 0 \end{bmatrix} \right) + W^{(i+1)} \begin{bmatrix} \mathcal{H}(-Z) b^{(i)} \\ 0 \end{bmatrix} + b^{(i+1)} \\ &= W^{(i+1)} f^{(i+1)}(x) + b^{(i+1)}. \end{aligned}$$

■

B.2 G-invariant architectures

Proof [Proof of Lemma 4] Since $f^{(1)}$ and $\psi^{(1)}$ are both the identity functions, then the claim is immediate for $i = 1$. Suppose the claim is true for some $i \in \{1, \dots, d-1\}$. We have for all $g \in G$ and $x \in \mathbb{R}^m$:

$$\begin{aligned}
f^{(i+1)}(gx) &= \begin{bmatrix} \mathrm{ReLU}(W^{(i)} f^{(i)}(gx) + b^{(i)}) \\ f^{(i)}(gx) \end{bmatrix} \\
&= \begin{bmatrix} \mathrm{ReLU}(W^{(i)} \psi^{(i)}(g) f^{(i)}(x) + b^{(i)}) \\ \psi^{(i)}(g) f^{(i)}(x) \end{bmatrix} \\
&= \begin{bmatrix} \mathrm{ReLU}(\rho^{(i)}(g) W^{(i)} f^{(i)}(x) + b^{(i)}) \\ \psi^{(i)}(g) f^{(i)}(x) \end{bmatrix} \\
&= \begin{bmatrix} \mathrm{ReLU}(\rho^{(i)}(g) W^{(i)} f^{(i)}(x) + \rho^{(i)}(g) b^{(i)}) \\ \psi^{(i)}(g) f^{(i)}(x) \end{bmatrix} \\
&= \begin{bmatrix} \pi^{(i)}(g)\, \mathrm{ReLU}(\zeta^{(i)}(g)(W^{(i)} f^{(i)}(x) + b^{(i)})) \\ \psi^{(i)}(g) f^{(i)}(x) \end{bmatrix} \\
&= \begin{bmatrix} \pi^{(i)}(g)\, \mathrm{ReLU}(W^{(i)} f^{(i)}(x) + b^{(i)}) \\ \psi^{(i)}(g) f^{(i)}(x) \end{bmatrix} - \begin{bmatrix} \pi^{(i)}(g) \mathcal{H}(-\zeta^{(i)}(g)) W^{(i)} f^{(i)}(x) \\ 0 \end{bmatrix} \\
&\quad - \begin{bmatrix} \pi^{(i)}(g) \mathcal{H}(-\zeta^{(i)}(g)) b^{(i)} \\ 0 \end{bmatrix} \\
&= \begin{bmatrix} \pi^{(i)}(g) & -\pi^{(i)}(g) \mathcal{H}(-\zeta^{(i)}(g)) W^{(i)} \\ 0 & I_{N_i} \end{bmatrix} f^{(i+1)}(x) + \begin{bmatrix} -\pi^{(i)}(g) \mathcal{H}(-\zeta^{(i)}(g)) b^{(i)} \\ 0 \end{bmatrix}.
\end{aligned}$$

Note that

$$\begin{aligned}
-\pi^{(i)}(g) \mathcal{H}(-\zeta^{(i)}(g)) &= -\frac{1}{2} \pi^{(i)}(g)(I_{n_{i+1}} - \zeta^{(i)}(g)) \\
&= \frac{1}{2} \pi^{(i)}(g)(\zeta^{(i)}(g) - I_{n_{i+1}}) \\
&= \frac{1}{2}(\rho^{(i)}(g) - \pi^{(i)}(g)).
\end{aligned}$$

We thus have

$$\begin{aligned}
-\pi^{(i)}(g) \mathcal{H}(-\zeta^{(i)}(g)) W^{(i)} &= \frac{1}{2}(\rho^{(i)}(g) - \pi^{(i)}(g)) W^{(i)} \\
&= \frac{1}{2}(\rho^{(i)}(g) W^{(i)} - \pi^{(i)}(g) W^{(i)}) \\
&= \frac{1}{2}(W^{(i)} \psi^{(i)}(g) - \pi^{(i)}(g) W^{(i)}).
\end{aligned}$$

For the bias term, let $\rho^{(i)} = \rho_1^{(i)} \oplus \cdots \oplus \rho_k^{(i)}$ be the decomposition of $\rho^{(i)}$ into irreducibles; decompose $\pi^{(i)}$ and $b^{(i)}$ correspondingly. Then we have

$$\begin{aligned} -\pi^{(i)}(g)\mathcal{H}(-\zeta^{(i)}(g))b^{(i)} &= \frac{1}{2}(\rho^{(i)}(g) - \pi^{(i)}(g))b^{(i)} \\ &= \frac{1}{2}\bigoplus_{j=1}^{k}(\rho_j^{(i)}(g) - \pi_j^{(i)}(g))b_j^{(i)}. \end{aligned}$$

If $\rho_j^{(i)}$ is type 1, then $\rho_j^{(i)} = \pi_j^{(i)}$ so that the jth term in the above direct sum is zero. On the other hand, if $\rho_j^{(i)}$ is type 2, then $\rho_j^{(i)}(g)b_j^{(i)} = b_j^{(i)}$ implies $b_j^{(i)} = 0$, so that again the jth summand is zero. Therefore, $-\pi^{(i)}(g)\mathcal{H}(-\zeta^{(i)}(g))b^{(i)} = 0 \forall g \in G$. We thus have

$$\begin{aligned} f^{(i+1)}(gx) &= \begin{bmatrix} \pi^{(i)}(g) & \frac{1}{2}(W^{(i)}\psi^{(i)}(g) - \pi^{(i)}(g)W^{(i)}) \\ 0 & I_{N_i} \end{bmatrix} f^{(i+1)}(x) + 0 \\ &= \psi^{(i+1)}(g)f^{(i+1)}(x). \end{aligned}$$

The conclusion follows by induction. ■

Proof [Proof of Lemma 5] We will verify that $\psi^{(i+1)}$ defined as claimed satisfies the recursion in Lemma 4. For $i = 1$, we have

$$\begin{aligned} \psi^{(2)}(g) &= A^{(1)-1}\Pi^{(1)}(g)A^{(1)} \\ &= \begin{bmatrix} I_{n_2} & -\frac{1}{2}W^{(1)} \\ 0 & I_{n_1} \end{bmatrix}^{-1} \begin{bmatrix} \pi^{(1)}(g) & 0 \\ 0 & g \end{bmatrix} \begin{bmatrix} I_{n_2} & -\frac{1}{2}W^{(1)} \\ 0 & I_{n_1} \end{bmatrix} \\ &= \begin{bmatrix} I_{n_2} & \frac{1}{2}W^{(1)} \\ 0 & I_{n_1} \end{bmatrix} \begin{bmatrix} \pi^{(1)}(g) & 0 \\ 0 & g \end{bmatrix} \begin{bmatrix} I_{n_2} & -\frac{1}{2}W^{(1)} \\ 0 & I_{n_1} \end{bmatrix} \\ &= \begin{bmatrix} \pi^{(1)}(g) & \frac{1}{2}(W^{(1)}g - \pi^{(1)}(g)W^{(1)}) \\ 0 & g \end{bmatrix} \\ &= \begin{bmatrix} \pi^{(1)}(g) & \frac{1}{2}(W^{(1)}\psi^{(1)}(g) - \pi^{(1)}(g)W^{(1)}) \\ 0 & \psi^{(1)}(g) \end{bmatrix}, \end{aligned}$$

which indeed agrees with Lemma 4.

Now suppose the claim holds for some $i \in \{2, \ldots, d-1\}$. Observe that

$$\begin{aligned} A^{(i)} &= \begin{bmatrix} I_{n_{i+1}} & -\frac{1}{2}W^{(i)} \\ 0 & A^{(i-1)} \end{bmatrix} \\ \Pi^{(i)}(g) &= \begin{bmatrix} \pi^{(i)}(g) & 0 \\ 0 & \Pi^{(i-1)}(g) \end{bmatrix}. \end{aligned}$$

We thus have

$$\begin{aligned}
\psi^{(i+1)}(g) &= A^{(i)-1}\Pi^{(i)}(g)A^{(i)} \\
&= \begin{bmatrix} I_{n_{i+1}} & -\frac{1}{2}W^{(i)} \\ 0 & A^{(i-1)} \end{bmatrix}^{-1} \begin{bmatrix} \pi^{(i)}(g) & 0 \\ 0 & \Pi^{(i-1)}(g) \end{bmatrix} \begin{bmatrix} I_{n_{i+1}} & -\frac{1}{2}W^{(i)} \\ 0 & A^{(i-1)} \end{bmatrix} \\
&= \begin{bmatrix} I_{n_{i+1}} & \frac{1}{2}W^{(i)}A^{(i-1)-1} \\ 0 & A^{(i-1)-1} \end{bmatrix} \begin{bmatrix} \pi^{(i)}(g) & 0 \\ 0 & \Pi^{(i-1)}(g) \end{bmatrix} \begin{bmatrix} I_{n_{i+1}} & -\frac{1}{2}W^{(i)} \\ 0 & A^{(i-1)} \end{bmatrix} \\
&= \begin{bmatrix} \pi^{(i)}(g) & \frac{1}{2}(W^{(i)}A^{(i-1)-1}\Pi^{(i-1)}(g)A^{(i-1)} - \pi^{(i)}(g)W^{(i)}) \\ 0 & A^{(i-1)-1}\Pi^{(i-1)}(g)A^{(i-1)} \end{bmatrix} \\
&= \begin{bmatrix} \pi^{(i)}(g) & \frac{1}{2}(W^{(i)}\psi^{(i)}(g) - \pi^{(i)}(g)W^{(i)}) \\ 0 & \psi^{(i)}(g) \end{bmatrix}.
\end{aligned}$$

The conclusion follows by induction. ■

Proof [Proof of Thm. 6] **(a)** By Lemmas 4-5, we have

$$\begin{aligned}
\rho^{(i)}(g)W^{(i)} &= W^{(i)}\psi^{(i)}(g) \\
\rho^{(i)}(g)W^{(i)} &= W^{(i)}A^{(i-1)-1}\Pi^{(i-1)}(g)A^{(i-1)} \\
\rho^{(i)}(g)W^{(i)}A^{(i-1)-1} &= W^{(i)}A^{(i-1)-1}\Pi^{(i-1)}(g).
\end{aligned}$$

Let $V^{(i)} = W^{(i)}A^{(i-1)-1}$. Thus, $W^{(i)} = V^{(i)}A^{(i-1)}$.

All that is left is to establish equivariance of the blocks of $V^{(i)}$. We have

$$\begin{aligned}
\rho^{(i)}(g)V^{(i)} &= \rho^{(i)}(g)W^{(i)}A^{(i-1)-1} \\
&= W^{(i)}\psi^{(i)}(g)A^{(i-1)-1} \\
&= W^{(i)}A^{(i-1)-1}\Pi^{(i-1)}(g)A^{(i-1)}A^{(i-1)-1} \\
&= V^{(i)}\Pi^{(i-1)}(g).
\end{aligned}$$

Since $\Pi^{(i-1)}(g)$ is a block-diagonal matrix, then blockwise equivariance follows.

(b) By part (a), we have

$$\begin{aligned}
g^{(i)}(x) &= W^{(i)}f^{(i)}(x) \\
&= V^{(i)}A^{(i-1)}f^{(i)}(x) \\
&= V^{(i)}h^{(i)}(x).
\end{aligned}$$

To establish the recursion for $h^{(i)}$, first observe that $A^{(i)}$ satisfies the recursion

$$A^{(i)} = \begin{bmatrix} I_{n_{i+1}} & -\frac{1}{2}W^{(i)} \\ 0 & A^{(i-1)} \end{bmatrix}.$$

We thus have

$$\begin{aligned} h^{(i+1)}(x) &= A^{(i)} f^{(i+1)}(x) \\ &= \begin{bmatrix} I_{n_{i+1}} & -\frac{1}{2}W^{(i)} \\ 0 & A^{(i-1)} \end{bmatrix} \begin{bmatrix} \mathrm{ReLU}(W^{(i)} f^{(i)}(x) + b^{(i)}) \\ f^{(i)}(x) \end{bmatrix} \\ &= \begin{bmatrix} \mathrm{ReLU}(W^{(i)} f^{(i)}(x) + b^{(i)}) - \frac{1}{2}W^{(i)} f^{(i)}(x) \\ A^{(i-1)} f^{(i)}(x) \end{bmatrix} \\ &= \begin{bmatrix} \mathrm{ReLU}(g^{(i)}(x) + b^{(i)}) - \frac{1}{2}g^{(i)}(x) \\ h^{(i)}(x) \end{bmatrix}, \end{aligned}$$

which completes the proof. ■

B.2.1 Implementation

Here we describe how Thm. 6 translates into a concrete implementation. First, independently for every $i \in \{1, \dots, d\}$ and $j \in \{1, \dots, i\}$, we construct a basis set $\mathcal{B}_j^{(i)}$ for the space of all matrix solutions to the linear system

$$\rho^{(i)}(g)X = X\pi^{(j-1)}(g) \forall g \in G.$$

This construction is done just once at the time of architecture initialization. We describe the construction below; for now, however, let us assume we have found such basis sets. Then we next constrain each latent weight matrix block $V_j^{(i)}$ to be a linear combination in $\mathcal{B}_j^{(i)}$. The linear coefficients are randomly initialized and are the trainable parameters of the G-DNN model. The forward pass is then a direct implementation of Thm. 6 (b), and its pseudocode is presented in Alg. 1.

Algorithm 1 Implementation of Thm. 6 (b) for the forward pass of a G-DNN f. Here $g^{(i)}(x)$ and $h^{(i)}(x)$ should be regarded as variable names.

function $f(x)$
  $h^{(1)}(x) \leftarrow x$
  for $i \in \{1, \dots, d-1\}$ **do**
    $g^{(i)}(x) \leftarrow V^{(i)} h^{(i)}(x)$
    $h^{(i+1)}(x) \leftarrow \begin{bmatrix} \mathrm{ReLU}(g^{(i)}(x) + b^{(i)}) - \frac{1}{2}g^{(i)}(x) \\ h^{(i)}(x) \end{bmatrix}$
  return $V^{(d)} g^{(d)}(x) + b^{(d)}$

All that remains is to describe the construction of the basis sets $\mathcal{B}_j^{(i)}$. We phrase this as the following general problem. Given a signed perm-rep ρ, an ordinary perm-rep π, and a generating set G_0 for the group G, we seek a basis set for the space of matrix solutions to the linear system

$$\rho(g)X\pi(g)^\top = X \forall g \in G_0.$$

While we could proceed with standard methods of numerical linear algebra, there is a risk that numerical instabilities in the computation of the rank could lead to an incorrect number

of basis matrices, which could break equivariance. We thus take a combinatorial approach to obtain an exact basis set as described next.

We construct a simple directed graph Γ to encode the (signed) permutation of matrix elements under the mapping $X \to \rho(g)X\pi(g)^\top \forall g \in G_0$. The nodes of Γ are the ordered pairs (i,j) indexing the elements in X, and we draw an arc from (i,j) to (k,ℓ) iff the former is sent to the latter under the above mapping for some $g \in G_0$; moreover, we assign the arc a value $z_{ij,k\ell} \in \{-1,1\}$ to indicate whether a sign flip is incurred. With the graph Γ in hand, a matrix X is a solution to the linear equivariance condition iff all constraints $x_{ij} = z_{ij,k\ell}x_{k\ell}$ are satisfied. We are thus able to assign node values x_{ij} independently across the connected components of Γ; based on this, we generate the desired basis set by processing each connected component C as follows:

Case 1. Suppose C is 2-colorable, meaning that each node $(i,j) \in C$ can be assigned a value $x_{ij} \in \{-1,1\}$ such that the constraints $x_{ij} = z_{ij,k\ell}x_{k\ell}$ are satisfied within C. Then we select such an assignment (by greedy 2-coloring), and we assign all nodes outside C the value zero. These values together yield a basis matrix.

Case 2. Suppose C is not 2-colorable. Then the only consistent assignment of values inside C is all zeros, and no basis matrix is returned for this component.

Observe that no two basis matrices share a nonzero value in the same position, and hence the matrices are indeed linearly independent.

B.3 Admissible architectures

B.3.1 The θ function

The following proposition establishes the invariance and equivariance of the function θ with respect to certain conjugations.

Proposition 8 *The function θ satisfies the following properties:*

(a) For every $A \in \mathrm{P}(|G/J|)$, define the ordinary perm-rep $\pi_J^A(g) = A\pi_J(g)A^{-1}$. Then $\theta(\cdot,\cdot,J)$ is invariant under the conjugation $\pi_J \mapsto \pi_J^A$.

(b) $\theta(H,K,gJg^{-1}) = \theta(H,K,J) \forall g \in G$.

(c) $\theta(gHg^{-1},gKg^{-1},J) = g\theta(H,K,J)g^{-1}$.

Proof (a) For brevity, let $\kappa = |H:K| - 1$. Define the function θ^A in identical fashion to θ, but replace π_J with π_J^A. We wish to prove $\theta^A = \theta$. Since the map $\Gamma \mapsto P_\Gamma$ from finite orthogonal matrix groups to orthogonal projection operators is equivariant with respect to conjugation, then we have

$$\begin{aligned}
\theta^A(H,K,J) &= \{g \in G : \pi_J^A(g)(P_{\pi_J^A(K)} - \kappa P_{\pi_J^A(H)}) = P_{\pi_J^A(K)} - \kappa P_{\pi_J^A(H)}\} \\
&= \{g \in G : A\pi_J(g)A^{-1}(P_{A\pi_J(K)A^{-1}} - \kappa P_{A\pi_J(H)A^{-1}}) = P_{A\pi_J(K)A^{-1}} - \kappa P_{A\pi_J(H)A^{-1}}\} \\
&= \{g \in G : A\pi_J(g)A^{-1}(AP_{\pi_J(K)}A^{-1} - \kappa AP_{\pi_J(H)}A^{-1}) = AP_{\pi_J(K)}A^{-1} \\
&\quad - \kappa AP_{\pi_J(H)}A^{-1}\} \\
&= \{g \in G : \pi_J(g)(P_{\pi_J(K)} - \kappa P_{\pi_J(H)}) = P_{\pi_J(K)} - \kappa P_{\pi_J(H)}\} \\
&= \theta(H,K,J).
\end{aligned}$$

(b) The theory of ordinary perm-irreps and their correspondence to group action on cosets is well-understood (Burnside, 1911; Bouc, 2000), and it is known that the conjugation of J in π_J is equivalent to the conjugation of π_J itself. The claim thus follows by part (a).

(c) Let $a \in G$. We will show $\theta(aHa^{-1}, aKa^{-1}, J) = \theta(H, K, J)$. Note $\kappa = |H : K| - 1$ is invariant under the conjugation of (H, K) by a. Also note $\pi_J(aHa^{-1}) = \pi_J(a)\pi_J(H)\pi_J(a)^{-1}$ and similar for K. Letting $A = \pi_J(a)$ and proceeding in analogy to part (a), we have

$$\begin{aligned}
\theta(aHa^{-1}, aKa^{-1}, J) &= \{g \in G : \pi_J(g)(P_{\pi_J(aKa^{-1})} - \kappa P_{\pi_J(aHa^{-1})}) = P_{\pi_J(aKa^{-1})} \\
&\quad - \kappa P_{\pi_J(aHa^{-1})}\} \\
&= \{g \in G : \pi_J(g)(P_{\pi_J^A(K)} - \kappa P_{\pi_J^A(H)}) = P_{\pi_J^A(K)} - \kappa P_{\pi_J^A(H)}\} \\
&= \{g \in G : \pi_J(g)(AP_{\pi_J(K)}A^{-1} - \kappa AP_{\pi_J(H)}A^{-1}) = AP_{\pi_J(K)}A^{-1} \\
&\quad - \kappa AP_{\pi_J(H)}A^{-1}\} \\
&= \{g \in G : A^{-1}\pi_J(g)A(P_{\pi_J(K)} - \kappa P_{\pi_J(H)}) = P_{\pi_J(K)} - \kappa P_{\pi_J(H)}\} \\
&= \{g \in G : \pi_J(a^{-1}ga)(P_{\pi_J(K)} - \kappa P_{\pi_J(H)}) = P_{\pi_J(K)} - \kappa P_{\pi_J(H)}\}.
\end{aligned}$$

By the change of variables $g \to aga^{-1}$, we have

$$\begin{aligned}
\theta(aHa^{-1}, aKa^{-1}, J) &= \{aga^{-1} \in G : \pi_J(g)(P_{\pi_J(K)} - \kappa P_{\pi_J(H)}) = P_{\pi_J(K)} - \kappa P_{\pi_J(H)}\} \\
&= a\{g \in G : \pi_J(g)(P_{\pi_J(K)} - \kappa P_{\pi_J(H)}) = P_{\pi_J(K)} - \kappa P_{\pi_J(H)}\}a^{-1} \\
&= a\theta(H, K, J)a^{-1},
\end{aligned}$$

establishing the claim. ■

Algorithm 2 Implementation of the function θ that exploits existing functions in GAP.

```
1:  function θ(H, K, J)
2:      def η : K \ G/J ↦ power(G/J) by
3:          η(KxJ) = {kxJ ∈ G/J : k(K ∩ xJx^{-1}) ∈ K/(K ∩ xJx^{-1})}
4:      if |H : K| = 2 then
5:          let h ∈ H \ K
6:          S ← {KxJ ∈ K \ G/J : hx ∈ KxJ}
7:      else
8:          S ← ∅
9:      if S = ∅ then
10:         T ← {η(KxJ) : KxJ ∈ K \ G/J}
11:     else
12:         T ← {η(KxJ) : KxJ ∈ (K \ G/J) \ S} ∪ {⋃_{KxJ∈S} η(KxJ)}
13:     return st_G(T)
```

Algorithm 2 gives the pseudocode for an implementation of the function θ that can be accomplished in the GAP language (GAP) for computational group theory. Although the definition of the θ function involves orthogonal projection operators, Alg. 2 completely circumvents these operators by taking a pure group-theoretic approach in terms of double cosets. The following proposition verifies that Alg. 2 is a correct implementation.

Proposition 9 *Algorithm 2 correctly implements the function θ.*

Proof Observe that the function η in Alg. 2 sends each double coset $KxJ \in K \setminus G/J$ to the set of cosets in G/J whose disjoint union is KxJ.

Let $H, K, J \leq G$ such that $K \leq H$ and $|H : K| \leq 2$. Let $w \in \text{ran}(P_{\pi_J(K)} - (|H : K| - 1)P_{\pi(H)})$. We regard w as a function $w : G/J \mapsto \mathbb{R}$. Since $w \in \text{ran}(P_K)$, then w is K-invariant in the sense that $w(kxJ) = w(xJ)$ for all $k \in K$ and $xJ \in G/J$. Thus, w is constant over the set $\eta(KxJ)$ for every $KxJ \in K \setminus G/J$.

If $|H : K| = 2$, then let $h \in H \setminus K$. Since $w \in \text{ran}(P_{\pi_J(K)} - P_{\pi_J(H)})$, then $w(hKxJ) = -w(KxJ)$ for every $KxJ \in K \setminus G/J$. Thus, if $hKxJ = KxJ$, then $w(KxJ) = 0$. Since $K \trianglelefteq H$, then $hKxJ = KxJ$ is equivalent to $KhxJ = KxJ$, or just $hx \in KxJ$. We thus have the constraint

$$w(KxJ) = 0 \forall KxJ \in K \setminus G/J \mid hx \in KxJ.$$

Now select w such that it takes a different nonzero value over each $\eta(KxJ)$ for all $KxJ \in K \setminus G/J$ such that, if $|H : K| = 2$, then $hx \notin KxJ$. Then

$$\begin{aligned}\theta(H, K, J) &= \text{st}_G(P_{\pi_J(K)} - (|H : K| - 1)P_{\pi_J(H)}) \\ &= \text{st}_G(w) \\ &= \{g \in G : w(gKxJ) = w(KxJ) \forall KxJ \in K \setminus G/J\}.\end{aligned}$$

This means $\theta(H, K, J)$ is exactly the subgroup of G that leaves the level sets of w invariant. Observe, however, that the sets in the collection T in Alg. 2 are exactly the level sets of w, and hence $\theta(H, K, J) = \text{st}_G(T)$. ■

B.3.2 The ϕ function

Proof [Proof of Prop. 7] The necessity of condition (2) is only because if $H_j^{(1)} = G$ for any j, then at least one row w of $W^{(1)}$ satisfies $w \in \text{ran}(P_G)$ by Thm. 4a of Agrawal and Ostrowski (2022). For w to be nonzero, we thus require $P_G \neq 0$. We assume condition (2) to be satisfied for the remainder of the proof.

Before proving the claim itself, we first derive a closed-form expression for $\phi^{(i+1)}$. For notational convenience, for every $K \leq H \leq G$, $|H : K| \leq 2$, and linear rep $\tau : G \mapsto \text{GL}(n, \mathbb{R})$, define

$$\theta(H, K; \tau) = \{g \in G : \tau(g)(P_{\tau(K)} - (|H : K| - 1)P_{\tau(H)}) = P_{\tau(K)} - (|H : K| - 1)P_{\tau(H)}\}.$$

Thus, $\theta(H, K, J)$ can be equivalently written as $\theta(H, K; \pi_J)$. For two reps τ_1 and τ_2, observe that $P_{\tau_1 \oplus \tau_2} = P_{\tau_1} \oplus P_{\tau_2}$ and hence

$$\theta(H, K; \tau_1 \oplus \tau_2) = \theta(H, K; \tau_1) \cap \theta(H, K; \tau_2).$$

This property extends to more than two reps in the obvious way. With this notation, and recalling the identity rep $\pi^{(0)} : G \mapsto G$, the function $\phi^{(i+1)}$ can be rewritten as

$$\begin{aligned}
\phi^{(i+1)}(H,K) &= \mathrm{st}_G(P_K - (|H:K|-1)P_H) \cap \bigcap_{j=1}^{i} \bigcap_{r=1}^{r^{(j)}} \theta(H,K,H_r^{(j)}) \\
&= \theta(H,K;\pi^{(0)}) \cap \bigcap_{j=1}^{i} \bigcap_{r=1}^{r^{(j)}} \theta(H,K;\pi_r^{(j)}) \\
&= \theta(H,K;\pi^{(0)}) \cap \theta\left(H,K;\bigoplus_{j=1}^{i} \bigoplus_{r=1}^{r^{(j)}} \pi_r^{(j)}\right) \\
&= \theta\left(H,K;\pi^{(0)} \oplus \bigoplus_{j=1}^{i} \pi^{(j)}\right) \\
&= \theta\left(H,K;\bigoplus_{j=0}^{i} \pi^{(j)}\right) \\
&= \theta(H,K;\Pi^{(i)}).
\end{aligned}$$

This is explicitly

$$\begin{aligned}
\phi^{(i+1)}(H,K) = \{g \in G : \Pi^{(i)}(g)(P_{\Pi^{(i)}(K)} - (|H:K|-1)P_{\Pi^{(i)}(H)}) = P_{\Pi^{(i)}(K)} \\
- (|H:K|-1)P_{\Pi^{(i)}(H)}\},
\end{aligned}$$

and it is equivalent to

$$\begin{aligned}
\phi^{(i+1)}(H,K) = \{g \in G : \Pi^{(i)}(g)(P_{\Pi^{(i)}(K)} - (|\Pi^{(i)}(H):\Pi^{(i)}(K)|-1)P_{\Pi^{(i)}(H)}) = P_{\Pi^{(i)}(K)} \\
- (|\Pi^{(i)}(H):\Pi^{(i)}(K)|-1)P_{\Pi^{(i)}(H)}\}. \qquad (*)
\end{aligned}$$

To see that $|\Pi^{(i)}(H):\Pi^{(i)}(K)| = |H:K|$, by the First Isomorphism Theorem we have

$$|\Pi^{(i)}(H):\Pi^{(i)}(K)| = \frac{|H|/|H \cap \ker(\Pi^{(i)})|}{|K|/|K \cap \ker(\Pi^{(i)})|}.$$

Since $\Pi^{(i)}$ includes in its direct sum decomposition the identity rep $\pi^{(0)}$, then $\ker(\Pi^{(i)})$ must be trivial, and so the above expression reduces to $|H|/|K| = |H:K|$.

We finally prove the claim. By definition, the G-DNN architecture $\{\rho^{(1)},\ldots,\rho^{(d)}\}$ is admissible iff each row of $W^{(i)}$ is nonzero and no two rows of $[W^{(i)} \mid b^{(i)}]$ are parallel, for $i \in \{1,\ldots,d\}$. By Thm. 6 (a), since $W^{(i)} = V^{(i)}A^{(i-1)}$, then we obtain an equivalent definition if we replace $W^{(i)}$ with $V^{(i)}$. Let $V_{(j)}^{(i)}$ be the submatrix comprising the rows (and all columns) of $V^{(i)}$ that together transform by $\rho_j^{(i)}$:

$$\rho_j^{(i)}(g)V_{(j)}^{(i)} = V_{(j)}^{(i)}\Pi^{(i-1)}(g) \forall g \in G.$$

(We include the parentheses in the subscript of $V_{(j)}^{(i)}$ to distinguish it from $V_j^{(i)}$ appearing in Thm. 6 (a)). Then Thm. 4a of Agrawal and Ostrowski (2022) implies we have an admissible architecture (specifically, that the rows of $V_{(j)}^{(i)}$ are nonzero and no two rows of the corresponding augmented weight matrix are parallel) iff

$$\mathrm{st}_{\Pi^{(i-1)}(G)}(P_{\Pi^{(i-1)}(K_j^{(i)})} - (|\Pi^{(i-1)}(H_j^{(i)}) : \Pi^{(i-1)}(K_j^{(i)})| - 1)P_{\Pi^{(i-1)}(H_j^{(i)})}) = \Pi^{(i-1)}(K_j^{(i)}),$$

or equivalently,

$$(\Pi^{(i-1)})^{-1}[\mathrm{st}_{\Pi^{(i-1)}(G)}(P_{\Pi^{(i-1)}(K_j^{(i)})} - (|\Pi^{(i-1)}(H_j^{(i)}) : \Pi^{(i-1)}(K_j^{(i)})| - 1)P_{\Pi^{(i-1)}(H_j^{(i)})}) = K_j^{(i)}$$

$$\{g \in G : \Pi^{(i-1)}(g)(P_{\Pi^{(i-1)}(K_j^{(i)})} - (|\Pi^{(i-1)}(H_j^{(i)}) : \Pi^{(i-1)}(K_j^{(i)})| - 1)P_{\Pi^{(i-1)}(H_j^{(i)})}) = P_{\Pi^{(i-1)}(K_j^{(i)})} - (|\Pi^{(i-1)}(H_j^{(i)}) : \Pi^{(i-1)}(K_j^{(i)})| - 1)P_{\Pi^{(i-1)}(H_j^{(i)})} = K_j^{(i)}.$$

Recalling (*), we recognize the last equation as nothing but $\phi^{(i)}(H_j^{(i)}, K_j^{(i)}) = K_j^{(i)}$, there proving that condition (1) is (together with condition (2)) is equivalent to admissibility. ■

B.4 Additional remarks

The following proposition establishes the compatibility of batchnorm with G-DNNs.

Proposition 10 *The addition of batchnorm immediately after any ReLU layer in a G-DNN preserves G-invariance of the network.*

Proof Suppose we apply batchnorm immediately after the ith ReLU layer of the G-DNN f, for some $i \in \{1, \ldots, d-1\}$. Then the activations $r^{(i)}(x) = \mathrm{ReLU}(W^{(i)} f^{(i)}(x) + b^{(i)})$ of the ith ReLU layer transform as

$$r^{(i)}(x) \to \gamma \left(\frac{r^{(i)}(x) - \mu \vec{1}}{\sigma + \varepsilon} \right) + \beta \vec{1},$$

where $\mu \geq 0$, $\sigma \geq 0$, $\varepsilon > 0$, γ, and β are all scalars. For each $i \in \{1, \ldots, d\}$, let $W_i^{(i)}$ and $V_i^{(i)}$ be the blocks comprising the first n_i columns of $W^{(i)}$ and $V^{(i)}$ respectively; these represent the weights of the ith layer without the skip connections. Then the above affine transformation of $r^{(i)}(x)$ is equivalent to the transformation

$$\begin{aligned} W_{i+1}^{(i+1)} &\to C W_{i+1}^{(i+1)} \\ b^{(i+1)} &\to b^{(i+1)} + D W_{i+1}^{(i+1)} \vec{1}, \end{aligned}$$

where

$$\begin{aligned} C &= \frac{\gamma}{\sigma + \varepsilon} \\ D &= \beta - \frac{\gamma \mu}{\sigma + \varepsilon}. \end{aligned}$$

By Thm. 6 (a), $W^{(i+1)} = V^{(i+1)}A^{(i)}$. Since $A^{(i)}$ is upper block-triangular with the top left block $I_{n_{i+1}}$, then $W_{i+1}^{(i+1)} = V_{i+1}^{(i+1)}$. By the above transformation of $W_{i+1}^{(i+1)}$ under batchnorm, $V_{i+1}^{(i+1)}$ also transforms only by the scalar factor C, and hence it remains equivariant as in Thm. 6a.

All that remains is to show the bias $b^{(i+1)}$ satisfies the sufficient condition in Lemma 4 even after the batchnorm transformation, and this will establish G-invariance of the network with batchnorm.

Case 1: Suppose $\rho^{(i+1)}$ is type 1 irreducible. Then Lemma 4 implies $b^{(i+1)}$ is parallel to $\vec{1}$. Under batchnorm, $b^{(i+1)}$ transforms by the addition of $DV_{i+1}^{(i+1)}\vec{1}$ and thus remains parallel to $\vec{1}$. The claim follows by Lemma 4.

Case 2: Suppose $\rho^{(i+1)}$ is type 2 irreducible. Then Lemma 4 implies $b^{(i+1)} = 0$. Under batchnorm, the bias thus transforms to $0 + DV_{i+1}^{(i+1)}\vec{1}$. It turns out, however, that $V_{i+1}^{(i+1)}\vec{1} = 0$; to see this, by Thm. 6 (a), we have

$$\rho^{(i+1)}(g)V_{i+1}^{(i+1)} = V_{i+1}^{(i+1)}\pi^{(i)}(g) \forall g \in G.$$

Since $i \geq 1$ so that $\pi^{(i)}$ is type 1, then without loss of generality, by selecting an appropriate basis, we assume $\pi^{(i)}$ is an ordinary perm-irrep. Averaging both sides over all $g \in G$, we obtain

$$P_{\rho^{(i+1)}}V_{i+1}^{(i+1)} = V_{i+1}^{(i+1)}P_{\pi^{(i)}}.$$

Since $\rho^{(i+1)}$ is type 2, then its only fixed point is the zero vector, and hence the orthogonal projection operator $P_{\rho^{(i+1)}}$ is itself zero. Moreover, since $\pi^{(i)}$ is an ordinary perm-irrep and since $\vec{1}$ is fixed under all permutations, then it is fixed under the orthogonal projection operator $P_{\pi^{(i)}}$ as well– hence $V_{i+1}^{(i+1)}\vec{1} = 0$.

Case 3: Suppose $\rho^{(i+1)}$ is reducible. Then decompose it into type1 and type 2 irreps and apply Cases 1-2 separately to each irrep. ■

The extension of Prop. 10 to multiple channels is trivial. Batchnorm is typically applied independently to each channel. Thus, if the ReLU activations $r^{(i)}(x)$ had c channels (which could be achieved by having c copies of every irrep in $\rho^{(i)}$), then the variables μ, σ, γ, and β would be c-dimensional vectors. The proof would then proceed by selecting a single arbitrary channel.

Appendix C. Examples

C.1 Concatenated ReLU

Proof [Proof of Ex. 1] For every $i \in \{1, \ldots, d\}$, the function

$$x \to U^{(i)}\,\mathrm{CReLU}(U^{(i-1)} \cdots \mathrm{CReLU}(U^{(1)}x) \cdots)$$

is G-equivariant with its output transforming by $\rho^{(i)}$. We thus identify it with the function $g^{(i)}$ appearing in Thm. 6 (b). We now have the recursion

$$g^{(i+1)}(x) = U^{(i+1)}\,\mathrm{CReLU}(g^{(i)}(x)) \forall i \in \{1, \ldots, d-1\}.$$

By definition of CReLU$(\cdot)$ and the block structure of $U^{(i+1)}$, we have

$$\begin{aligned}
g^{(i+1)}(x) &= \begin{bmatrix} U_1^{(i+1)} & U_2^{(i+1)} \end{bmatrix} \mathrm{ReLU}\left(\begin{bmatrix} g^{(i)}(x) \\ -g^{(i)}(x) \end{bmatrix} \right) \\
&= U_1^{(i+1)} \mathrm{ReLU}(g^{(i)}(x)) + U_2^{(i+1)} \mathrm{ReLU}(-g^{(i)}(x)) \\
&= U_1^{(i+1)} \mathrm{ReLU}(g^{(i)}(x)) + U_2^{(i+1)} \mathrm{ReLU}(g^{(i)}(x)) - U_2^{(i+1)} g^{(i)}(x) \\
&= (U_1^{(i+1)} + U_2^{(i+1)}) \mathrm{ReLU}(g^{(i)}(x)) - U_2^{(i+1)} g^{(i)}(x) \\
&= (U_1^{(i+1)} + U_2^{(i+1)}) \mathrm{ReLU}(g^{(i)}(x)) - \frac{1}{2}(U_1^{(i+1)} + U_2^{(i+1)}) g^{(i)}(x) + \frac{1}{2}(U_1^{(i+1)} - U_2^{(i+1)}) g^{(i)}(x) \\
&= \begin{bmatrix} U_1^{(i+1)} + U_2^{(i+1)} & \frac{1}{2}(U_1^{(i+1)} - U_2^{(i+1)}) \end{bmatrix} \begin{bmatrix} \mathrm{ReLU}(g^{(i)}(x)) - \frac{1}{2} g^{(i)}(x) \\ g^{(i)}(x) \end{bmatrix}.
\end{aligned}$$

Since $g^{(i)}(x) = V^{(i)} h^{(i)}(x)$ in Thm. 6 (b) and since all bias vectors in this example are zero, then we can write the above as

$$g^{(i+1)}(x) = \begin{bmatrix} U_1^{(i+1)} + U_2^{(i+1)} & \frac{1}{2}(U_1^{(i+1)} - U_2^{(i+1)}) V^{(i)} \end{bmatrix} \begin{bmatrix} \mathrm{ReLU}(g^{(i)}(x) + b^{(i)}) - \frac{1}{2} g^{(i)}(x) \\ h^{(i)}(x) \end{bmatrix}.$$

Comparing this to the definitions of $g^{(i+1)}$ and $h^{(i+1)}$ appearing in Thm. 6 (b), we obtain the expression for $V^{(i+1)}$ as claimed. ■

C.2 Binary multiplication

Proof [Proof of Ex. 2] **(a)** For $x_1, x_2 \in \{-1, 1\}$, it can be verified by hand that

$$\begin{aligned}
x_1 x_2 &= \begin{bmatrix} 1 & -1 \end{bmatrix} \mathrm{ReLU}\left(\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right) - x_2 \\
&= \begin{bmatrix} 1 & -1 & 0 & -1 \end{bmatrix} \begin{bmatrix} \mathrm{ReLU}\left(\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right) \\ \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \end{bmatrix}. (*)
\end{aligned}$$

We can extend this to the product of more than two elements as follows: For $i \in \{1, \ldots, d\}$, define the block-diagonal matrices

$$\begin{aligned}
A^{(i)} &= I_{m/2^i} \begin{bmatrix} 1 & -1 \end{bmatrix} \\
B^{(i)} &= I_{m/2^i} \begin{bmatrix} 0 & -1 \end{bmatrix} \\
C^{(i)} &= I_{m/2^i} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix},
\end{aligned}$$

and define the functions $p^{(i)} : \mathcal{X} \mapsto \{-1, 1\}^{m/2^i}$ by

$$\begin{aligned}
p^{(1)}(x) &= A^{(1)} x \\
p^{(i)}(x) &= \begin{bmatrix} A^{(i)} & B^{(i)} \end{bmatrix} \begin{bmatrix} \mathrm{ReLU}(C^{(i)} p^{(i-1)}(x)) \\ p^{(i-1)}(x) \end{bmatrix} \forall i \in \{2, \ldots, d\}. (**)
\end{aligned}$$

The function $p^{(1)}$ maps each pair of elements (x_{2i-1}, x_{2i}) in the input x to $x_{2i-1} - x_{2i} \in \{-1, 1\}$. Then $p^{(i)}$ (1) partitions $p^{(1)}(x)$ into blocks, each of two elements; (2) takes the product of each pair of elements using (*); (3) iterates this procedure $i-1$ times. The output $p^{(d)}(x)$ is then the product of the elements in $p^{(1)}(x)$.

Now for $i \in \{1, \ldots, d-1\}$, the function $g^{(i)}$ (as in Thm. 6 (b)) as given by $g^{(i)}(x) = C^{(i+1)}p^{(i)}(x)$. By (**), we thus have

$$\begin{aligned}
g^{(i)}(x) &= \begin{bmatrix} C^{(i+1)}A^{(i)} & C^{(i+1)}B^{(i)} \end{bmatrix} \begin{bmatrix} \mathrm{ReLU}(g^{(i-1)}(x)) \\ p^{(i-1)}(x) \end{bmatrix} \\
&= \begin{bmatrix} C^{(i+1)}A^{(i)} & C^{(i+1)}B^{(i)} \end{bmatrix} \begin{bmatrix} \mathrm{ReLU}(g^{(i-1)}(x)) - \frac{1}{2}g^{(i-1)}(x) + \frac{1}{2}g^{(i-1)}(x) \\ p^{(i-1)}(x) \end{bmatrix} \\
&= \begin{bmatrix} C^{(i+1)}A^{(i)} & C^{(i+1)}B^{(i)} \end{bmatrix} \begin{bmatrix} \mathrm{ReLU}(g^{(i-1)}(x)) - \frac{1}{2}g^{(i-1)}(x) \\ p^{(i-1)}(x) \end{bmatrix} \\
&\quad + \begin{bmatrix} C^{(i+1)}A^{(i)} & C^{(i+1)}B^{(i)} \end{bmatrix} \begin{bmatrix} \frac{1}{2}g^{(i-1)}(x) \\ 0 \end{bmatrix} \\
&= \begin{bmatrix} C^{(i+1)}A^{(i)} & C^{(i+1)}B^{(i)} \end{bmatrix} \begin{bmatrix} \mathrm{ReLU}(g^{(i-1)}(x)) - \frac{1}{2}g^{(i-1)}(x) \\ p^{(i-1)}(x) \end{bmatrix} + \frac{1}{2}C^{(i+1)}A^{(i)}g^{(i-1)}(x) \\
&= \begin{bmatrix} C^{(i+1)}A^{(i)} & C^{(i+1)}B^{(i)} \end{bmatrix} \begin{bmatrix} \mathrm{ReLU}(g^{(i-1)}(x)) - \frac{1}{2}g^{(i-1)}(x) \\ p^{(i-1)}(x) \end{bmatrix} \\
&\quad + \frac{1}{2}C^{(i+1)}A^{(i)}C^{(i)}p^{(i-1)}(x) \\
&= \begin{bmatrix} C^{(i+1)}A^{(i)} & C^{(i+1)}B^{(i)} + \frac{1}{2}C^{(i+1)}A^{(i)}C^{(i)} \end{bmatrix} \begin{bmatrix} \mathrm{ReLU}(g^{(i-1)}(x)) - \frac{1}{2}g^{(i-1)}(x) \\ p^{(i-1)}(x) \end{bmatrix} \\
&= \begin{bmatrix} C^{(i+1)}A^{(i)} & 0 \end{bmatrix} \begin{bmatrix} \mathrm{ReLU}(g^{(i-1)}(x)) - \frac{1}{2}g^{(i-1)}(x) \\ p^{(i-1)}(x) \end{bmatrix} \\
&= \begin{bmatrix} C^{(i+1)}A^{(i)} & 0 \end{bmatrix} \begin{bmatrix} \mathrm{ReLU}(g^{(i-1)}(x)) - \frac{1}{2}g^{(i-1)}(x) \\ h^{(i-1)}(x) \end{bmatrix}.
\end{aligned}$$

Comparing this to the equations in Thm. 6 (b), we establish the claimed expression for $V_j^{(i)}$ for $i \in \{1, \ldots, d-1\}$.

For the case $i = d$, we simply observe that the last weight matrix must be the outer weight vector in (*). The G-invariance of the constructed DNN is clear; the action of any $g \in G$ on an input $x \in \mathcal{X}$ corresponds to an even number of sign flips in $p^{(1)}(x)$, which leaves the parity of the product $p^{(d)}(x)$ invariant. Layerwise G-equivariance is established next.

(b) By part (a), The first weight matrix is

$$C^{(2)}A^{(1)} = I_{m/4} \otimes \begin{bmatrix} 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}.$$

The jth pair of rows in this weight matrix is the jth channel, whose first row is v_j and which transforms by $\rho_{H_j^{(1)}K_j^{(1)}}$. The expressions for $H_j^{(1)}$ and $K_j^{(1)}$ are thus established by definition.

Now for $i \in \{1, \ldots, d\}$, part (a) implies

$$\rho_{H_j^{(i+1)} K_j^{(i+1)}}(g) \begin{bmatrix} 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} \pi_{H_{2j-1}^{(i)}}(g) & 0 \\ 0 & \pi_{H_{2j}^{(i)}}(g) \end{bmatrix} \forall g \in G,$$

where each $\pi_{H_k^{(i)}}$ is the ordinary perm-rep part of $\rho_{H_k^{(i)} K_k^{(i)}}$. By definition, $K_j^{(i+1)}$ is the subgroup of all $g \in G$ such that

$$\begin{aligned} \rho_{H_j^{(i+1)} K_j^{(i+1)}}(g) e_1 &= e_1 \\ \frac{1}{2} \rho_{H_j^{(i+1)} K_j^{(i+1)}}(g) \begin{bmatrix} 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} &= e_1 \\ \frac{1}{2} \begin{bmatrix} 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} \pi_{H_{2j-1}^{(i)}}(g) & 0 \\ 0 & \pi_{H_{2j}^{(i)}}(g) \end{bmatrix} \forall g \in G \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} &= e_1. \end{aligned}$$

Since ordinary perm-reps cannot flip signs, then the last equation is equivalent to

$$\pi_{H_{2j-1}^{(i)}} e_1 = e_1 \text{ and } \pi_{H_{2j}^{(i)}} e_1 = e_1.$$

We thus have

$$\begin{aligned} K_j^{(i+1)} &= \{g \in G : \pi_{H_{2j-1}^{(i)}} e_1 = e_1\} \cap \{g \in G : \pi_{H_{2j}^{(i)}} e_1 = e_1\} \\ &= H_{2j-1}^{(i)} \cap H_{2j}^{(i)}. \end{aligned}$$

Similarly, by definition we have

$$H_j^{(i+1)} = \{g \in G : \rho_{H_j^{(i+1)} K_j^{(i+1)}}(g) e_1 = \pm e_1\}.$$

Proceeding analogously as above, we find that $g \in G$ is contained in $H_j^{(i+1)}$ iff

$$(\pi_{H_{2j-1}^{(i)}}(g) e_1 = e_1 \text{ and } \pi_{H_{2j}^{(i)}}(g) e_1 = e_1) \text{ or } (\pi_{H_{2j-1}^{(i)}}(g) e_1 = e_2 \text{ and } \pi_{H_{2j}^{(i)}}(g) e_1 = e_2).$$

The first term in the disjunction corresponds to $H_{2j-1}^{(i)} \cap H_{2j}^{(i)}$, and the second term in the disjunction corresponds to the intersection of the complements, as claimed.

In the case of the $(d-1)$st layer, the product output $p^{(d-1)}(x)$ is 2D and thus can only transform by ± 1. The rep $\rho^{(d-)}$ thus decomposes into two copies of a scalar rep. Finally, that the final rep $\rho^{(d)}$ is trivial follows from the G-invariance of the network, thereby completing the proof. ■

References

Devanshu Agrawal and James Ostrowski. A classification of G-invariant shallow neural networks. *Advances in Neural Information Processing Systems*, 35, 2022.

Devanshu Agrawal, Adrian Del Maestro, Steven Johnston, and James Ostrowski. Group-equivariant autoencoder for identifying spontaneously broken symmetries. *Physical Review E*, 107(5):054104, 2023.

Kenneth Atz, Francesca Grisoni, and Gisbert Schneider. Geometric deep learning on molecular representations. *Nature Machine Intelligence*, 3(12):1023–1032, 2021.

Michael Baake. Structure and representations of the hyperoctahedral group. *Journal of mathematical physics*, 25(11):3171–3182, 1984.

David Balduzzi, Marcus Frean, Lennox Leary, JP Lewis, Kurt Wan-Duo Ma, and Brian McWilliams. The shattered gradients problem: If resnets are the answer, then what is the question? In *International Conference on Machine Learning*, pages 342–350. PLMR, 2017.

Sourya Basu, Akshayaa Magesh, Harshit Yadav, and Lav R Varshney. Autoequivariant network search via group decomposition. *arXiv preprint arXiv:2104.04848*, 2021.

Serge Bouc. Burnside rings. In *Handbook of algebra*, volume 2, pages 739–804. Elsevier, 2000.

Michael M Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

William Burnside. *Theory of groups of finite order*. The University Press, 1911.

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

Taco S. Cohen and Max Welling. Steerable cnns. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017. URL `https://openreview.net/forum?id=rJQKYt5ll`.

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

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

Marc Finzi, Max Welling, and Andrew Gordon Wilson. A practical method for constructing equivariant multilayer perceptrons for arbitrary matrix groups. In *International conference on machine learning*, pages 3318–3328. PLMR, 2021.

GAP. GAP – Groups, Algorithms, and Programming, Version 4.11.1. https://www.gap-system.org, 2021.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

Chiyu Jiang, Jingwei Huang, Karthik Kashinath, Prabhat, Philip Marcus, and Matthias Nießner. Spherical cnns on unstructured grids. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019. URL https://openreview.net/forum?id=Bkl-43C9FQ.

Oumar Kaba and Siamak Ravanbakhsh. Equivariant networks for crystal structures. *Advances in Neural Information Processing Systems*, 35:4150–4164, 2022.

Piotr Kicki, Mete Ozay, and Piotr Skrzypczyński. A computationally efficient neural network invariant to the action of symmetry subgroups. *arXiv preprint arXiv:2002.07528*, 2020.

Piotr Kicki, Piotr Skrzypczyński, and Mete Ozay. A new approach to design symmetry invariant neural networks. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2021.

Risi Kondor and Shubhendu Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. In *International Conference on Machine Learning*, pages 2747–2755. PLMR, 2018.

Di Luo, Giuseppe Carleo, Bryan K Clark, and James Stokes. Gauge equivariant neural networks for quantum lattice gauge theories. *Physical review letters*, 127(27):276402, 2021.

Kaitlin Maile, Dennis George Wilson, and Patrick Forré. Equivariance-aware architectural optimization of neural networks. In *The Eleventh International Conference on Learning Representations*, 2022.

Vincent Mallet and Jean-Philippe Vert. Reverse-complement equivariant networks for dna sequences. *Advances in Neural Information Processing Systems*, 34:13511–13523, 2021.

Haggai Maron, Heli Ben-Hamu, Nadav Shamir, and Yaron Lipman. Invariant and equivariant graph networks. volume 7, 2019a.

Haggai Maron, Ethan Fetaya, Nimrod Segol, and Yaron Lipman. On the universality of invariant networks. In *International conference on machine learning*, pages 4363–4371. PMLR, 2019b.

Mohamed Adel Musallam, Vincent Gaudilliere, Miguel Ortiz Del Castillo, Kassem Al Ismaeil, and Djamila Aouada. Leveraging equivariant features for absolute pose regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6876–6886, 2022.

Siamak Ravanbakhsh. Universal equivariant multilayer perceptrons. In *International Conference on Machine Learning*, pages 7996–8006. PMLR, 2020.

Siamak Ravanbakhsh, Jeff Schneider, and Barnabas Poczos. Equivariance through parameter-sharing. In *International conference on machine learning*, pages 2892–2901. PLMR, 2017.

Wenling Shang, Kihyuk Sohn, Diogo Almeida, and Honglak Lee. Understanding and improving convolutional neural networks via concatenated rectified linear units. In *international conference on machine learning*, pages 2217–2225. PMLR, 2016.

Wenling Shang, Justin Chiu, and Kihyuk Sohn. Exploring normalization in deep residual networks with concatenated rectified linear units. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

Bastiaan S Veeling, Jasper Linmans, Jim Winkens, Taco Cohen, and Max Welling. Rotation equivariant cnns for digital pathology. In *International Conference on Medical image computing and computer-assisted intervention*, pages 210–218. Springer, 2018.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

Hannah Zhou, Avanti Shrikumar, and Anshul Kundaje. Towards a better understanding of reverse-complement equivariance for deep learning models in genomics. In *Machine Learning in Computational Biology*, pages 1–33. PMLR, 2022.