Learning Probabilistic Symmetrization for Architecture Agnostic Equivariance

Jinwoo Kim **Tien Dat Nguyen** **Ayhan Suleymanzade**
Hyeokjun An **Seunghoon Hong**
KAIST

Abstract

We present a novel framework to overcome the limitations of equivariant architectures in learning functions with group symmetries. In contrary to equivariant architectures, we use an arbitrary base model (such as an MLP or a transformer) and symmetrize it to be equivariant to the given group by employing a small equivariant network that parameterizes the probabilistic distribution underlying the symmetrization. The distribution is end-to-end trained with the base model which can maximize performance while reducing sample complexity of symmetrization. We show that this approach ensures not only equivariance to given group but also universal approximation capability in expectation. We implement our method on a simple patch-based transformer that can be initialized from pretrained vision transformers, and test it for a wide range of symmetry groups including permutation and Euclidean groups and their combinations. Empirical tests show competitive results against tailored equivariant architectures, suggesting the potential for learning equivariant functions for diverse groups using a non-equivariant universal base architecture. We further show evidence of enhanced learning in symmetric modalities, like graphs, when pretrained from non-symmetric modalities, like vision. Our implementation will be open-sourced at `https://github.com/jw9730/lps`.

1 Introduction

Many perception problems in machine learning involve functions that are invariant or equivariant to certain symmetry group of transformations of data. Examples include learning on sets and graphs, point clouds, molecules, proteins, and physical data, to name a few [10, 12]. Equivariant architecture design has emerged as a successful approach, where every building block of a model is carefully restricted to be equivariant to a symmetry group of interest [6, 12, 23]. However, equivariant architecture design faces fundamental limitations, as individual construction of models for each group can be laborious or computationally expensive [81, 56, 60, 43], the architectural restrictions often lead to limited expressive power [89, 54, 93, 38], and the knowledge learned from one problem cannot be easily transferred to others of different symmetries as the architecture would be incompatible.

This motivates us to seek a **symmetrization** solution that can achieve group invariance and equivariance with general-purpose, group-agnostic architectures such as an MLP or a transformer. As a basic form of symmetrization, any parameterized function $f_\theta : \mathcal{X} \to \mathcal{Y}$ on vector spaces $\mathcal{X}, \mathcal{Y}$ can be made invariant or equivariant by group averaging [90, 63], *i.e.*, averaging over all possible transformations of inputs $\mathbf{x} \in \mathcal{X}$ and outputs $\mathbf{y} \in \mathcal{Y}$ by a symmetry group $G = \{g\}$:

$$\phi_\theta(\mathbf{x}) = \frac{1}{|G|} \sum_{g \in G} g \cdot f_\theta(g^{-1} \cdot \mathbf{x}), \tag{1}$$

where ϕ_θ is equivariant or invariant to the group G. An important advantage is that the symmetrized function ϕ_θ can leverage the expressive power of the base function f_θ; it has been shown that ϕ_θ is a universal approximator of invariant or equivariant functions if f_θ is a universal approximator [90], which includes an MLP [34] or a transformer [92]. On the other hand, an immediate challenge is that

Preprint. Under review.

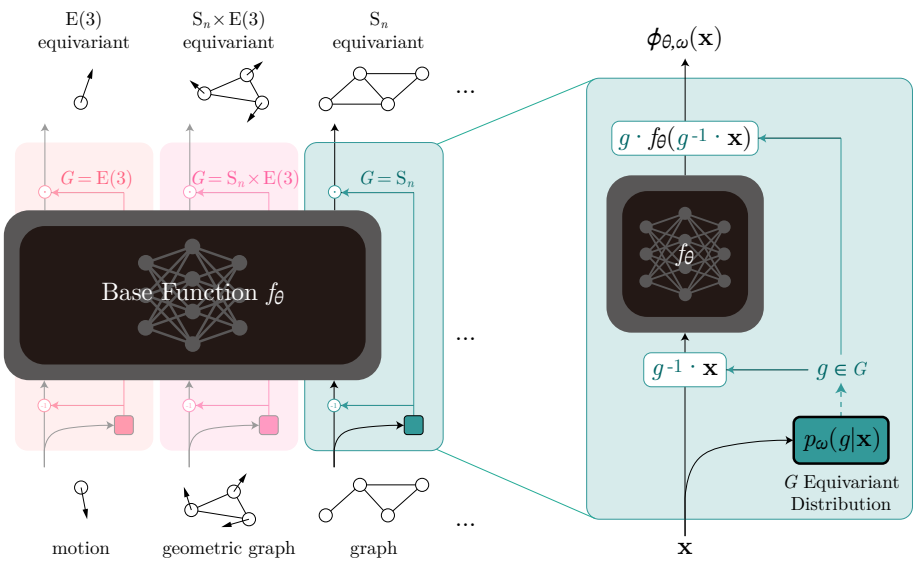

Figure 1: Overview of probabilistic symmetrization. We symmetrize an unconstrained base function f_θ into an equivariant function $\phi_{\theta,\omega}$ for group G using a learned equivariant distribution $p_\omega(g|\mathbf{x})$.

for many practical groups involving permutation and rotations, the cardinality of the group $|G|$ is large or infinite, so the exact averaging is intractable. Due to this, existing symmetrization approaches often focus on small finite groups [5, 61, 83, 40], manually derive smaller subsets of the entire group to average over [69, 63, 64], or implement a relaxed version of equivariance [39, 75].

An alternative method for tractability is to interpret Eq. (1) as an expectation with uniform distribution $\mathrm{Unif}(G)$ over the compact group G [58], and use sampling-based average to estimate it [90, 63, 64]:

$$\phi_\theta(\mathbf{x}) = \mathbb{E}_{g \sim \mathrm{Unif}(G)} \left[g \cdot f_\theta(g^{-1} \cdot \mathbf{x}) \right], \tag{2}$$

where $g \cdot f_\theta(g^{-1} \cdot \mathbf{x})$ serves as an unbiased estimator of $\phi_\theta(\mathbf{x})$. While simple and general, this approach has practical issues that the base function f_θ is burdened to learn all equally possible group transformations, and the expectedly high variance of the estimator can lead to challenges in sampling-based training due to large variance of gradients as well as sample complexity of inference.

Our key idea is to replace the uniform distribution $\mathrm{Unif}(G)$ for the expectation in Eq. (2) with a parameterized distribution $p_\omega(g|\mathbf{x})$ in a way that equivariance and expressive power are always guaranteed, and train it end-to-end with the base function f_θ to directly minimize task loss. We show that the distribution $p_\omega(g|\mathbf{x})$ only needs to satisfy one simple condition to guarantee equivariance and expressive power: it has to be probabilistically equivariant [9]. This allows us to generally implement $p_\omega(g|\mathbf{x})$ as a noise-outsourced map $(\mathbf{x}, \boldsymbol{\epsilon}) \mapsto g$ with an invariant noise $\boldsymbol{\epsilon}$ and a small equivariant network q_ω, which enables gradient-based training with reparameterization [45]. As p_ω is trained, it can enhance the learning of f_θ by producing group transformations of lower variance compared to $\mathrm{Unif}(G)$ so that f_θ is less burdened, and coordinating with f_θ to maximize task performance. We refer to our approach as probabilistic symmetrization. An overview is provided in Figure 1.

We implement and test our method with two general-purpose architectures as the base function f_θ: MLP and transformer. In particular, our transformer backbone is architecturally identical to patch-based vision transformers [26], which allows us to initialize most of its parameters from ImageNet-21k pretrained weights [3] and only replace the input and output projections to match task dimensions. We implement the conditional distribution $p_\omega(g|\mathbf{x})$ for a wide range of practical symmetry groups including permutation (S_n) and Euclidean groups ($O/SO(d)$ and $E/SE(d)$) and their product combinations (*e.g.*, $S_n \times O(3)$), all of which are combinatorial or infinite groups. Empirical tests on a wide range of invariant and equivariant tasks involving graphs and motion data show competitive results against tailored equivariant architectures as well as existing symmetrization methods [69, 39]. This suggests the potential for learning invariant or equivariant functions for diverse groups with a group-agnostic general-purpose backbone. We further show evidence that pretraining from non-symmetric modality (vision) leads to enhanced learning in symmetric modality (graphs).

2 Probabilistic Symmetrization for Equivariance

We introduce and analyze our approach called probabilistic symmetrization in Section 2.1, which involves an equivariant distribution p_ω and group-agnostic base function f_θ. We then describe implementation of p_ω for practical groups including permutations and rotations in Section 2.2. Then, we describe our choice of base function f_θ based on MLP and, transformers in particular, in Section 2.3. All proofs can be found in Appendix A.1.

Problem Setup In general, our goal is to construct a function $\phi : \mathcal{X} \to \mathcal{Y}$ on finite vector spaces $\mathcal{X}, \mathcal{Y}$ that is invariant or equivariant to symmetry specified by a group $G = \{g\}$[1]. This is formally described by specifying how the group act as transformations on the input and output. A group representation $\rho : G \to \mathrm{GL}(\mathcal{X})$, where $\mathrm{GL}(\mathcal{X})$ is the set of all invertible matrices on $\mathcal{X}$, associates each group element $g \in G$ to an invertible matrix $\rho(g)$ that transforms a given vector $\mathbf{x} \in \mathcal{X}$ through $\mathbf{x} \mapsto g \cdot \mathbf{x} = \rho(g)\mathbf{x}$. Given that, a function $\phi : \mathcal{X} \to \mathcal{Y}$ is G equivariant if:

$$\phi(\rho_1(g)\mathbf{x}) = \rho_2(g)\phi(\mathbf{x}), \quad \forall \mathbf{x} \in \mathcal{X}, g \in G, \tag{3}$$

where the representations ρ_1 and ρ_2 are on the input and output, respectively. G invariance is a special case of equivariance when the output representation is trivial, $\rho_2(g) = \mathbf{I}$.

2.1 Probabilistic Symmetrization

To construct a G equivariant function ϕ_θ, group averaging symmetrizes an arbitrary base function $f_\theta : \mathcal{X} \to \mathcal{Y}$ by taking expectation with uniform distribution over the group (Eq. (2)). Instead, we propose to use an input-conditional parameterized distribution $p_\omega(g|\mathbf{x})$ and symmetrize f_θ as follows:

$$\phi_{\theta,\omega}(\mathbf{x}) = \mathbb{E}_{p_\omega(g|\mathbf{x})}\left[\rho_2(g) f_\theta(\rho_1(g)^{-1}\mathbf{x})\right], \tag{4}$$

where the distribution $p_\omega(g|\mathbf{x})$ itself satisfies probabilistic G equivariance:

$$p_\omega(g|\mathbf{x}) = p_\omega(g'g|\rho_1(g')\mathbf{x}), \quad \forall \mathbf{x} \in \mathcal{X}, g, g' \in G. \tag{5}$$

Importantly, we show that probabilistic symmetrization with equivariant p_ω guarantees equivariance as well as expressive power of the symmetrized $\phi_{\theta,\omega}$.

Theorem 1. *If p_ω is G equivariant, then $\phi_{\theta,\omega}$ is G equivariant for arbitrary f_θ.*

Proof. The proof can be found in Appendix A.1.1. □

Theorem 2. *If p_ω is G equivariant and f_θ is a universal approximator, then $\phi_{\theta,\omega}$ is a universal approximator of G equivariant functions.*

Proof. The proof can be found in Appendix A.1.2. □

While the base function f_θ that guarantees universal approximation can be chosen in a group-agnostic manner *e.g.*, an MLP [34, 20] or a transformer on token sequences [92], the distribution p_ω needs to be instantiated group specifically to satisfy G equivariance. A simplistic choice is using uniform distribution $\mathrm{Unif}(G)$ for all inputs $\mathbf{x}$ with no parameterization (reducing to group averaging), which is technically equivariant and therefore guarantees equivariance and universality. However, appropriately parameterizing and learning p_ω can provide distinguished advantages, as it can **(1)** learn to collaborate with (pre-trained) base function f_θ to maximize task performance and **(2)** learn to produce more consistent samples $g \sim p_\omega(g|\mathbf{x})$ that can offer more stable gradients for the base function f_θ during early training.

We now provide a generic blueprint of G equivariant distribution $p_\omega(g|\mathbf{x})$ for any compact group G. Our goal is to sample $g \sim p_\omega(g|\mathbf{x})$ to obtain group representation $\rho(g)$ for symmetrization (Eq. (4)) in a differentiable manner so that p_ω can be trained end-to-end. Since we only need sampling and there is no need to evaluate likelihoods, we simply implement $p_\omega(g|\mathbf{x})$ as a noise-outsourced, differentiable transformation $q_\omega(\mathbf{x}, \boldsymbol{\epsilon})$ of a noise variable $\boldsymbol{\epsilon} \in \mathcal{E}$ that directly outputs a group representation $\rho(g)$:

$$\rho(g) = q_\omega(\mathbf{x}, \boldsymbol{\epsilon}), \quad \boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon}), \tag{6}$$

[1] In this paper, we assume the group G to be compact.

where q_ω is G equivariant and $p(\boldsymbol{\epsilon})$ is G invariant under a faithful orthogonal representation ρ':

$$q_\omega(\rho_1(g)\mathbf{x}, \rho'(g)\boldsymbol{\epsilon}) = \rho(g)q_\omega(\mathbf{x}, \boldsymbol{\epsilon}), \quad p(\boldsymbol{\epsilon}) = p(\rho'(g)\boldsymbol{\epsilon}), \quad \forall \mathbf{x} \in \mathcal{X}, \boldsymbol{\epsilon} \in \mathcal{E}, g \in G. \tag{7}$$

Given above implementation, we can show the G equivariance of p_ω:

Theorem 3. *If q_ω is G equivariant and $p(\boldsymbol{\epsilon})$ is G invariant under a faithful orthogonal group representation ρ', then the distribution $p_\omega(g|\mathbf{x})$ characterized by $q_\omega : (\mathbf{x}, \boldsymbol{\epsilon}) \mapsto \rho(g)$ is G equivariant.*

Proof. The proof can be found in Appendix A.1.3. □

In practice, one can use any available G equivariant neural network to implement q_ω, *e.g.*, a graph neural network for the symmetric group S_n, or an equivariant MLP which can be constructed for any matrix group [30]. Since we expect most of the reasoning to be done by the base function f_θ, the equivariant network q_ω can be small and relatively less expressive. This allows us to get less affected by their known issues in expressiveness and scaling [89, 66, 14, 38]. For the noise $\boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon})$, simple choices often suffice for G invariance. For example, standard normal $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}_n)$ provides invariance for the symmetric group S_n as well as the (special) orthogonal group $\mathrm{O}(n)$ and $\mathrm{SO}(n)$.

One important detail in designing q_ω is constraining its output to be a valid group representation $\rho(g)$. For this, we apply appropriate postprocessing to refine neural network features into group representations, *e.g.*, Gram-Schmidt orthogonalization to obtain a representation $\rho(g) \in \mathbb{R}^{n\times n}$ of the orthogonal group $g \in \mathrm{O}(n)$. Importantly, to not break the G equivariance of q_ω, this postprocessing needs to be equivariant itself, *e.g.*, Gram-Schmidt process is itself $\mathrm{O}(n)$ equivariant [39]. Implementations of p_ω for a range of practical symmetry groups are provided in detail in Section 2.2.

2.2 Equivariant Distribution p_ω

We present implementations of the G equivariant distribution $p_\omega(g|\mathbf{x})$ for a range of practical groups demonstrated in our experiments (Section 3). Formal proofs of correctness are in Appendix A.1.4.

Symmetric Group S_n The symmetric group S_n over a finite set of n elements contains all permutations of the set, which describes symmetry to ordering desired for learning set and graph data. The base representation is given by $\rho(g) = \mathbf{P}_g$ where $\mathbf{P}_g \in \{0, 1\}^{n\times n}$ is a permutation matrix for g.

To implement the S_n equivariant distribution $p_\omega(g|\mathbf{x})$ that provides permutation matrices $\mathbf{P}_g$ from a graph data $\mathbf{x}$, we use the following design. We first sample invariant noise $\boldsymbol{\epsilon} \in \mathbb{R}^{n\times d}$ from i.i.d. uniform $\mathrm{Unif}[0, \eta]$ with noise scale η. For the S_n equivariant map $q_\omega : (\mathbf{x}, \boldsymbol{\epsilon}) \mapsto \rho(g)$, we first use a graph neural network (GNN) as an equivariant map $(\mathbf{x}, \boldsymbol{\epsilon}) \mapsto \mathbf{Z}$ that outputs nodewise scalar $\mathbf{Z} \in \mathbb{R}^n$. Then, assuming $\mathbf{Z}$ is tie-free[2], we use below argsort operator [86] to postprocess $\mathbf{Z} \mapsto \mathbf{P}_g$:

$$\mathbf{P}_g \approx \hat{\mathbf{P}}_g = \mathrm{softmax}\left(\frac{-\|\mathbf{Z}\mathbf{1}^\top - \mathbf{1}\mathrm{sort}(\mathbf{Z})^\top\|_1}{\tau}\right), \tag{8}$$

where softmax applies to each row, $\|\cdot\|_1$ is L1 norm, and $\tau \in \mathbb{R}_+$ is a temperature hyperparameter. For an end-to-end training, we use straight-through estimator [7] with $\hat{\mathbf{P}}_g$ to differentiate through $\mathbf{P}_g$. The whole postprocessing is S_n equivariant, *i.e.*, it maps $\mathbf{P}_{g'}\mathbf{Z} \mapsto \mathbf{P}_{g'}\mathbf{P}_g$ for all $\mathbf{P}_{g'} \in \mathrm{S}_n$.

Orthogonal Group $\mathrm{O}(n)$, $\mathrm{SO}(n)$ The orthogonal group $\mathrm{O}(n)$ contains all roto-reflections in $\mathbb{R}^n$ around origin, and the special orthogonal group $\mathrm{SO}(n)$ contains all rotations without reflections. These groups describe rotation symmetries desirable in learning geometric data. The base group representation for $\mathrm{O}(n)$ is given by $\rho(g) = \mathbf{Q}_g$ where $\mathbf{Q}_g$ is the orthogonal matrix for g. For $\mathrm{SO}(n)$, the representation is $\rho(g) = \mathbf{Q}_g^+$ where $\mathbf{Q}_g^+$ is the orthogonal matrix of g with determinant $+1$.

To implement $\mathrm{O}(n)$/$\mathrm{SO}(n)$ equivariant distribution $p_\omega(g|\mathbf{x})$ that provides orthogonal matrices given input data $\mathbf{x}$, we use the following design. We first sample invariant noise $\boldsymbol{\epsilon} \in \mathbb{R}^{n\times d}$ from i.i.d. normal $\mathcal{N}(0, \eta^2)$ with noise scale η. For the $\mathrm{O}(n)$/$\mathrm{SO}(n)$ equivariant map $p_\omega : (\mathbf{x}, \boldsymbol{\epsilon}) \mapsto \rho(g)$, we first use an equivariant neural network as a map $(\mathbf{x}, \boldsymbol{\epsilon}) \mapsto \mathbf{Z}$ that outputs n features $\mathbf{Z} \in \mathbb{R}^{n\times n}$. Then, assuming $\mathbf{Z}$ is full-rank[3], we use Gram-Schmidt process for orthogonalization, which is differentiable

[2]This can be assumed since the invariant noise $\boldsymbol{\epsilon} \in \mathbb{R}^{n\times d}$ serves as tiebreaker between the n nodes.

[3]In practice, we add a random Gaussian matrix of a tiny magnitude to $\mathbf{Z}$ to prevent rank collapse.

and $\mathrm{O}(n)$ equivariant [39]. This completes the postprocessing $\mathbf{Z} \mapsto \mathbf{Q}_g$ for $\mathrm{O}(n)$. For $\mathrm{SO}(n)$, we further use simple scale operator $\mathbf{Q} \mapsto \mathbf{Q}_g^+$ to set the determinant of the orthogonalized matrix to $+1$:

$$\text{scale} : \left[\begin{array}{c|c|c} \mathbf{Q}_1 & \cdots & \mathbf{Q}_n \end{array}\right] \mapsto \left[\begin{array}{c|c|c} \det(\mathbf{Q}) \cdot \mathbf{Q}_1 & \cdots & \mathbf{Q}_n \end{array}\right], \tag{9}$$

The scale operator is differentiable and $\mathrm{SO}(n)$ equivariant, *i.e.*, it maps $\mathbf{Q}_{g'}^+ \mathbf{Q} \mapsto \mathbf{Q}_{g'}^+ \mathbf{Q}_g^+$ for all $\mathbf{Q}_{g'}^+ \in \mathrm{SO}(n)$, thereby completing the postprocessing $\mathbf{Z} \mapsto \mathbf{Q}_g^+$ for $\mathrm{SO}(n)$.

Euclidean Group $\mathrm{E}(n)$, $\mathrm{SE}(n)$ The Euclidean group $\mathrm{E}(n)$ contains all roto-translations and reflections in $\mathbb{R}^n$ and their combinations, and the special Euclidean group $\mathrm{SE}(n)$ contains all roto-translations without reflections. These groups are desired in learning physical systems such as a particle in motion. Formally, the Euclidean group is given as a combination of orthogonal group and translation group $\mathrm{E}(n) = \mathrm{O}(n) \ltimes \mathrm{T}(n)$, and similarly $\mathrm{SE}(n) = \mathrm{SO}(n) \ltimes \mathrm{T}(n)$. As the translation group $\mathrm{T}(n)$ is non-compact which violates our assumption for symmetrization, we handle it separately. Following prior work [69, 39], we subtract the centroid $\bar{\mathbf{x}}$ from the input data $\mathbf{x}$ as $\mathbf{x} - \bar{\mathbf{x}}$, and add it to the rotation symmetrized output as $\bar{\mathbf{x}} + g \cdot f_\theta(g^{-1} \cdot (\mathbf{x} - \bar{\mathbf{x}}))$ where g is sampled from $\mathrm{O}(n)$/$\mathrm{SO}(n)$ equivariant $p_\omega(g|\mathbf{x} - \bar{\mathbf{x}})$. This makes the overall symmetrized function $\mathrm{E}(n)$/$\mathrm{SE}(n)$ equivariant.

Product Group $H \times K$ While we have described several groups individually, in practice we often encounter product combinations of groups $G = H \times K$ that describe joint symmetry to each group H and K. For example, $\mathrm{S}_n \times \mathrm{O}(3)$ describes joint symmetry to permutations and rotations, which is desired in learning point clouds, molecules, and particle interactions. In general, an element of $H \times K$ is given as $g = (h, k)$ where $h \in H$ and $k \in K$, and group operations are applied elementwise $gg' = (h, k)(h', k') = (hh', kk')$. The base group representation is accordingly given as pair of representations $\rho(g) = (\rho(h), \rho(k))$. While a common approach to handling $H \times K$ is *partially* symmetrizing on H and imposing K equivariance on the base architecture (*e.g.*, rotational symmetrization of graph neural networks for $\mathrm{S}_n \times \mathrm{O}(3)$ equivariance [69, 39]), we choose *full symmetrization* on $H \times K$ since our goal is not imposing any constraint on the base function f_θ.

To implement $H \times K$ equivariant distribution $p_\omega(g|\mathbf{x})$ that gives $\rho(g) = (\rho(h), \rho(k))$ from data $\mathbf{x}$, we use the following design. We first sample invariant noise $\boldsymbol{\epsilon}$ from i.i.d. normal $\mathcal{N}(0, \eta^2)$ with scale η. For the $H \times K$ equivariant map $q_\omega : (\mathbf{x}, \boldsymbol{\epsilon}) \mapsto (\rho(h), \rho(k))$, we employ a $H \times K$ equivariant neural network as a map $(\mathbf{x}, \boldsymbol{\epsilon}) \mapsto (\mathbf{Z}_H, \mathbf{Z}_K)$ such that the postprocessing for each group H and K provides maps $\mathbf{Z}_H \mapsto \rho(h)$ and $\mathbf{Z}_K \mapsto \rho(k)$ respectively, leading to full representation $\rho(g) = (\rho(h), \rho(k))$. For this whole procedure to be $H \times K$ equivariant, it is sufficient to have $\mathbf{Z}_H$ be K invariant and $\mathbf{Z}_K$ be H invariant. This is a special case of $H \times K$ equivariance, and is supported by a range of equivariant neural networks especially regarding $\mathrm{S}_n \times \mathrm{O}(3)$ or $\mathrm{S}_n \times \mathrm{SO}(3)$ equivariances [23].

2.3 Base Function f_θ

We now describe the choice of group-agnostic base function $f_\theta : \mathbf{x} \mapsto \mathbf{y}$. As group symmetry is handled by the equivariant distribution $p_\omega(g|\mathbf{x})$, any symmetry concern is *hidden* from f_θ, allowing the inputs $\mathbf{x}$ and outputs $\mathbf{y}$ to be treated as plain multidimensional arrays. This allows us to implement f_θ with powerful general-purpose architectures, namely as an MLP that operates on flattened vectors of inputs and outputs, or a transformer as we describe below.

Let inputs $\mathbf{x} \in \mathcal{X} = \mathbb{R}^{n_1 \times \ldots \times n_a \times c}$ and outputs $\mathbf{y} \in \mathcal{Y} = \mathbb{R}^{n_1' \times \ldots \times n_b' \times c'}$ be multidimensional arrays with c and c' channels, respectively. Our transformer base function $f_\theta : \mathbf{x} \mapsto \mathbf{y}$ is given as:

$$f_\theta = \text{detokenize} \circ \text{transformer} \circ \text{tokenize}, \tag{10}$$

where $\text{tokenize} : \mathcal{X} \to \mathbb{R}^{m \times d}$ parses input array to a sequence of m tokens, $\text{transformer} : \mathbb{R}^{m \times d} \to \mathbb{R}^{m \times d}$ is a standard transformer encoder on tokens used in language and vision [24, 26], and $\text{detokenize} : \mathbb{R}^{m \times d} \to \mathcal{Y}$ decodes encoded tokens to output array. For the tokenizer and detokenizer, we use linear projections on flattened chunks of the array, which directly extends flattened patch projections in vision transformers to higher dimensions [26, 77, 78]. This allows us to map between different dimensional inputs and outputs, *e.g*, for graph node classification with $\mathbf{x} \in \mathbb{R}^{n \times n \times c}$ and $\mathbf{y} \in \mathbb{R}^{n \times c'}$, we use 2D patch projection for the input and 1D projection for the output.

Above choice of f_θ offers important advantages including universal approximation [92] and ability to share and transfer the learned knowledge in θ over different domains of different group symmetries. Remarkably, this allows us to directly leverage large-scale pre-trained parameters from data-abundant domains for learning on symmetric domains. In our experiments, we only replace the tokenizer and detokenizer of a vision transformer pre-trained on ImageNet-21k [87, 26] and fine-tune it to perform diverse S_n equivariant tasks such as graph classification, node classification, and link prediction.

2.4 Relation to Other Symmetrization Approaches

We discuss relation of our symmetrization method to prior ones, specifically group averaging [90, 5], frame averaging [69], and canonicalization [39]. An extended discussion on broader related work can be found in Appendix A.2. Since these symmetrization methods share common formalization $\mathbf{y} = \mathbb{E}_g[g \cdot f_\theta(g^{-1} \cdot \mathbf{x})]$, one can expect a close theoretical relationship between them. We observe that probabilistic symmetrization is quite general; based on particular choices of the G equivariant distribution $p_\omega(g|\mathbf{x})$, it can become most of the related symmetrization methods as special cases. This can be easily seen for group averaging [90], as the distribution p_ω can reduce to the uniform distribution $\mathrm{Unif}(G)$ over the group. Frame averaging [69] also takes an average, but over a subset of group given by a frame $F : \mathcal{X} \to 2^G \setminus \emptyset$; importantly, it is required that the frame itself is G equivariant $F(\rho(g)\mathbf{x}) = gF(\mathbf{x})$. We can make the following connection between our method and frame averaging, by adopting the concept of stabilizer subgroup $G_\mathbf{x} = \{g \in G : \rho(g)\mathbf{x} = \mathbf{x}\}$:

Proposition 1. *Probabilistic symmetrization with G equivariant distribution $p_\omega(g|\mathbf{x})$ can become frame averaging [69] by assigning uniform density to a set of orbits $G_\mathbf{x}g$ for some group elements g.*

Proof. The proof can be found in Appendix A.1.5. □

Canonicalization [39] uses a *single* group element for symmetrization, produced by a trainable canonicalizer $C_\omega : \mathcal{X} \to G$. Here, it is required that the canonicalizer itself satisfies *relaxed* G equivariance $C(\rho(g)\mathbf{x}) = gg'C(\mathbf{x})$ up to arbitrary action from the stabilizer $g' \in G_\mathbf{x}$. We now show:

Proposition 2. *Probabilistic symmetrization with G equivariant distribution $p_\omega(g|\mathbf{x})$ can become canonicalization [39] by assigning uniform density to a single orbit $G_\mathbf{x}g$ of some group element g.*

Proof. The proof can be found in Appendix A.1.5. □

Assuming that stabilizer $G_\mathbf{x}$ is trivial, this can be implemented with our method by removing random noise $\boldsymbol{\epsilon}$, which reduces p_ω to deterministic map $\rho(g) = q_\omega(\mathbf{x})$. We use this approach to implement canonicalizer for the S_n group, while [39] only provides canonicalizers for Euclidean groups.

3 Experiments

We empirically demonstrate and analyze probabilistic symmetrization on a wide range of symmetry groups S_n, $\mathrm{O}(3)$, $\mathrm{SO}(3)$, $\mathrm{E}(3)$, $\mathrm{SE}(3)$, and their products including $\mathrm{S}_n \times \mathrm{O}(3)$, on a variety of invariant and equivariant tasks, with general-purpose base functions f_θ chosen as MLP and transformer optionally with large-scale pretraining from computer vision domain. Details of the datasets and models used in the experiments can be found in Appendix A.3.

3.1 Graph Isomorphism Learning with MLP

Building expressive neural networks for graphs (S_n) has been considered important and challenging, as simple and efficient GNNs are often limited in expressive power to certain Weisfeiler-Lehman isomorphism tests like 1-WL [89, 54]. Since an MLP equipped with probabilistic symmetrization is in theory universal and S_n equivariant, it has potential for graph learning that require high expressive power. To explicitly test this, we adopt the experimental setup of [69] and use two datasets on graph separation task (S_n invariant). GRAPH8c [2] consists of all non-isomorphic connected graphs with 8 nodes, and EXP [1] consists of 3-WL distinguishable graphs that are not 2-WL distinguishable. We compare our method to standard GNNs as well as an MLP symmetrized with group averaging [90], frame averaging [69], and canonicalization [39]. Our method uses the same MLP architecture to symmetrization baselines, and its S_n equivariant distribution p_ω for symmetrization is implemented

Table 1: Results for S_n invariant graph separation. We use two tasks, one for counting pairs of graphs not separated by a model at random initialization (GRAPH8c and EXP), and one for learning to classify EXP to two classes (EXP-classify). For EXP-classify, we report the test accuracy at best validation accuracy. The columns arch. and sym. denote architectural and symmetrized equivariance, respectively. The results for baselines are from [69] except for MLP-Canonical. which is tested by us.

method	arch.	sym.	GRAPH8c ↓	EXP ↓	EXP-classify ↑
GCN [46]	S_n	-	4755	600	50%
GAT [85]	S_n	-	1828	600	50%
GIN [89]	S_n	-	386	600	50%
ChebNet [22]	S_n	-	44	71	82%
PPGN [54]	S_n	-	0	0	**100%**
GNNML3 [2]	S_n	-	0	0	**100%**
MLP-GA [90]	-	S_n	0	0	50%
MLP-FA [69]	-	S_n	0	0	**100%**
MLP-Canonical.	-	S_n	0	0	62%
MLP-PS (Ours)	-	S_n	0	0	**100%**

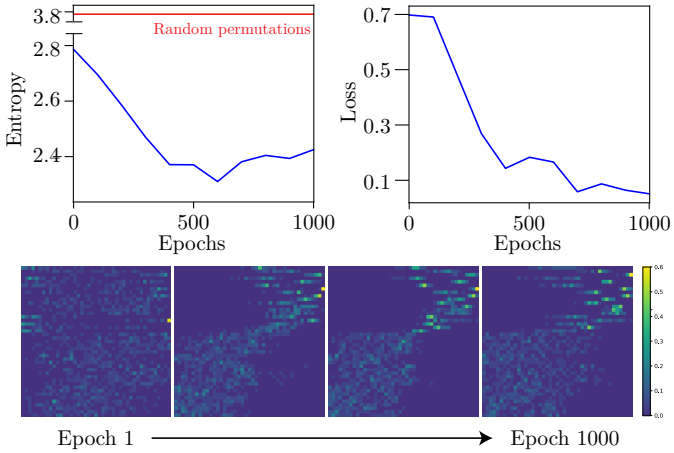

Figure 2: Learned $p_\omega(g|\mathbf{x})$ over time. The entropy of aggregated permutation matrices $\bar{\mathbf{P}} = \sum \mathbf{P}_g/N$ from $\mathbf{P}_g \sim p_\omega(g|\mathbf{x})$ for each input $\mathbf{x}$ drops in early training, indicating lower-variance sampled permutation as in below visualizations. Then the entropy increases slightly, but validation task loss still improves, indicating that p_ω and f_θ are learning to utilize stochasticity for better performance.

using a 3-layer GIN [89] which is 1-WL expressive. We use 10 samples for symmetrization during both training and testing. Further details of the experiment can be found in Appendix A.3.3.

The results are in Table 1. At random initialization, all symmetrization methods can provide perfect separation of all graphs, similar to PPGN [54] and GNNML3 [2] that are equivariant neural networks carefully designed to be 3-WL expressive. However, when trained with gradient descent to solve classification problem, naïve symmetrization with group averaging fails, presumably because the MLP fails to adjust to equally possible $64!$ permutations of 64 nodes in maximum. On the other hand, our method is able to learn the task, achieving the same accuracy to frame averaging that utilizes costly eigendecomposition of graph Laplacian [69]. What makes our method work while group averaging fails? We conjecture this is since the distribution $p_\omega(g|\mathbf{x})$ can learn to provide more consistent permutations during early training, as we illustrate in Figure 2. In the figure, we measured the consistency of samples from $p_\omega(g|\mathbf{x})$ over training progress by sampling $N = 50$ permutation matrices $\mathbf{P}_g \sim p_\omega(g|\mathbf{x})$ and measuring the row-wise entropy of their average $\bar{\mathbf{P}} = \sum \mathbf{P}_g/N$ for each input $\mathbf{x}$. The more consistent the sampled permutations, the sharper their average, and lesser the entropy. As training progresses, p_ω learns to produce more consistent samples, which coincides with the initial increase in task performance. Given that, a natural question would be: if we enforce the samples $g \sim p_\omega(g|\mathbf{x})$ to be consistent from the first place, would it work? To answer this, we also tested a non-probabilistic version of our model that uses a single permutation per input $\rho(g) = q_\omega(\mathbf{x})$, which is a canonicalizer under relaxed equivariance [39] as described in Section 2.4. As in Table 1, the canonicalization variant is unable to learn the task, indicating that the probabilistic nature of $p_\omega(g|\mathbf{x})$ is *beneficial* for early optimization.

Table 2: Results for $\mathrm{S}_n \times \mathrm{E}_n$ equivariant n-body problem. The columns arch. and sym. denote architectural and symmetrized equivariance, respectively. We report test MSE at best validation MSE, along with the standard deviation for GA and Ours where predictions are stochastic. The results for baselines are from [39] except symmetrized transformers which are tested by us.

method	arch.	sym.	Position MSE ↓
SE(3) Transformer [31]	$\mathrm{S}_n \times \mathrm{SE}(3)$	-	0.0244
TFN [81]	$\mathrm{S}_n \times \mathrm{SE}(3)$	-	0.0155
Radial Field [47]	$\mathrm{S}_n \times \mathrm{E}(3)$	-	0.0104
EGNN [76]	$\mathrm{S}_n \times \mathrm{E}(3)$	-	0.0071
GNN-FA [69]	S_n	$\mathrm{E}(3)$	0.0057
GNN-Canonical. [39]	S_n	$\mathrm{E}(3)$	0.0043
Transformer-Canonical.	-	$\mathrm{S}_n \times \mathrm{E}(3)$	0.00779
Transformer-GA	-	$\mathrm{S}_n \times \mathrm{E}(3)$	0.00419 ± 0.00001
Transformer-PS (Ours)	-	$\mathrm{S}_n \times \mathrm{E}(3)$	$\mathbf{0.00417 \pm 0.00002}$

3.2 Particle Dynamics Learning with Transformer

Learning sets or graphs attributed with position and velocity in 3D ($\mathrm{S}_n \times \mathrm{E}(3)$) is practically significant as they universally appear in physics, chemistry, and biology applications. While prior symmetrization methods employ an already S_n equivariant base function and partially symmetrize the $\mathrm{E}(3)$ part, we attempt to symmetrize the entire product group $\mathrm{S}_n \times \mathrm{E}(3)$ and choose the base model f_θ as a sequence transformer to leverage its expressive power. For empirical demonstration, we adopt the experimental setup of [39] and use the n-body dataset [76, 31] where the task is predicting the position of $n = 5$ charged particles after certain time given their initial position and velocity in $\mathbb{R}^3$ ($\mathrm{S}_n \times \mathrm{E}(3)$ equivariant). We compare our method to $\mathrm{S}_n \times \mathrm{E}(3)$ equivariant neural networks and partial symmetrization methods applying $\mathrm{E}(3)$ symmetrization to GNNs. We also test prior symmetrization methods on the full group $\mathrm{S}_n \times \mathrm{E}(3)$ along our method, but could not test for frame averaging since equivariant frames for the full group $\mathrm{S}_n \times \mathrm{E}(3)$ was not available in current literature. Our method is implemented using a transformer with sequence positional encodings with similar parameters to the baselines (around $1.2\times$), and the $\mathrm{S}_n \times \mathrm{E}(3)$ equivariant distribution p_ω for symmetrization is implemented using a 2-layer Vector Neurons [23] that has around $0.14\times$ of parameters to the transformer. We use 20 samples for symmetrization during training, and use $10\times$ sample size for testing since the task is regression where appropriate variance reduction is necessary to guarantee a reliable performance. Further details of the experiment can be found in Appendix A.3.4.

The results are in Table 2. We observe simple group averaging exhibits a surprisingly strong performance, as it achieves 0.00419 MSE and already outperforms previous state of the art 0.0043 MSE. This is because the permutation component of the symmetry is fairly small, with $n = 5$ particles interacting with each other, such that combining it with an expressive base model f_θ (a sequence transformer) can adjust to $5! = 120$ equally possible permutations and their rotations in 3D. Nevertheless, our method outperforms group averaging slightly and achieves a new state of the art 0.00417 MSE, presumably as the parameterized distribution p_ω learns to further maximize task performance. On the other hand, the canonicalization approach, implemented by eliminating noise from our method (Section 2.4), performs poorly. We empirically observe that f_θ memorizes the per-input canonical orientations provided by $\rho(g) = q_\omega(\mathbf{x})$ that do not generalize to test inputs. This again shows that probabilistic nature of $p_\omega(g|\mathbf{x})$ can be beneficial for performance.

3.3 Graph Pattern Recognition with Vision Transformer

One important goal of our approach, and symmetrization in general, is to *decouple* the symmetry of problem from the base function f_θ, such that we can leverage knowledge learned from other symmetries by transferring the parameters θ. We demonstrate an extreme case by transferring the parameters of a vision transformer [26] trained on large-scale image classification (translation invariant) to solve node classification on graphs (S_n equivariant) for the first time in literature. For this, we use the PATTERN dataset [27] that contains 14,000 purely topological random SBM graphs with 44-188 nodes, whose task is finding certain subgraph pattern by binary node classification.

Table 3: Results for S_n equivariant node classification on PATTERN. We report test accuracy at the best validation accuracy, along with the standard deviation for GA and Ours where predictions are stochastic. The results for GNN baselines are from [27].

method	pretrain.	Accuracy ↑
GCN [46], 16 layers	-	85.614
GAT [85], 16 layers	-	78.271
GatedGCN [11], 16 layers	-	85.568
GIN [89], 16 layers	-	85.387
RingGNN [16], 2 layers	-	86.245
RingGNN [16], 8 layers	-	diverged
PPGN [54], 3 layers	-	85.661
PPGN [54], 8 layers	-	diverged
ViT-GA, 1-sample	-	65.696 ± 0.063
ViT-GA, 10-sample	-	79.950 ± 0.058
ViT-GA, 1-sample	ImageNet-21k	82.412 ± 0.046
ViT-GA, 10-sample	ImageNet-21k	84.660 ± 0.036
ViT-FA	-	68.894
ViT-FA	ImageNet-21k	74.485
ViT-Canonical.	-	85.980
ViT-Canonical.	ImageNet-21k	86.048
ViT-PS (Ours), 1-sample	-	85.917 ± 0.037
ViT-PS (Ours), 10-sample	-	86.150 ± 0.032
ViT-PS (Ours), 1-sample	ImageNet-21k	86.299 ± 0.016
ViT-PS (Ours), 10-sample	ImageNet-21k	$\mathbf{86.353 \pm 0.023}$

Based on pre-trained ViT [26], we construct the base model f_θ by replacing only the input and output layers to take the flattened patches of 2D zero-padded adjacency matrices of size 188×188 and produce output as 1D per-node classification logits of length 188 with 2 channels. In addition to standard GNNs[4], we compare group averaging, frame averaging, and canonicalization to our method, and also test whether pre-trained representations from ImageNet-21k is beneficial for the task. For the S_n equivariant distribution p_ω in our method and canonicalization, we use a 3-layer GIN [89] with only 0.38% of the base model parameters. Further details can be found in Appendix A.3.5.

The results are in Table 3. First, we observe that transferring the pre-trained ViT parameters consistently improves the node classification for all symmetrization methods. It indicates that some traits of the pre-trained visual representation can benefit learning graph tasks which vastly differ in both the underlying symmetry (translation invariance $\rightarrow$ S_n equivariance) and the data generating process (natural images $\rightarrow$ random process of SBM). In particular, it is somewhat surprising that vision pretraining allows group averaging to achieve 84.660% accuracy, on par with GNN baselines, considering that memorizing all $188!$ equally possible permutations in this dataset is impossible. We conjecture that group averaged ViT can in some way learn meaningful graph representation internally to solve the task, and vision pretraining helps in acquiring the representation by providing a good initialization point or transferable computation motifs.

On the other hand, frame averaging shows a low performance, 74.485% accuracy with vision pretraining, which is also surprising considering that frames vastly reduce the sample space of symmetrization in general; in fact, the size of frame of each graph in PATTERN is exactly 1. We empirically observe that, unlike group averaging, ViT with frame averaging memorizes frames of training graphs rather than learning generalizable graph representations. In contrast, canonicalization that also uses a single sample per graph successfully learns the task with 86.048% accuracy. We conjecture that the learnable orderings provided by an equivariant neural network $\rho(g) = q_\omega(\mathbf{x})$ is more flexible and generalizable to unseen graphs compared to frames computed from fixed graph Laplacian eigenvectors. Lastly, our method achieves consistently better performance compared to other symmetrization methods, and the performance further improves with vision pretraining and more samples for testing. As a result, our model based on pre-trained ViT and 10 samples for testing achieves 86.353% test accuracy, surpassing all baselines.

[4]We note that careful engineering such as graph positional encoding improves performance of GNNs, but we have chosen simple and representative GNNs in the benchmark [27] to provide a controlled comparison.

Table 4: Results for real-world graph tasks. We report test performance at best validation performance.

method	Peptides-func	Peptides-struct	PCQM-Contact			
	AP ↑	MAE ↓	Hits@1 ↑	Hits@3 ↑	Hits@10 ↑	MRR ↑
GCN [46]	0.5930	0.3496	0.1321	0.3791	0.8256	0.3234
GCNII [15]	0.5543	0.3471	0.1325	0.3607	0.8116	0.3161
GINE [35]	0.5498	0.3547	0.1337	0.3642	0.8147	0.3180
GatedGCN [11]	0.5864	0.3420	0.1279	0.3783	0.8433	0.3218
GatedGCN+RWSE [11]	0.6069	0.3357	0.1288	0.3808	0.8517	0.3242
Transformer+LapPE [28]	0.6326	0.2529	0.1221	0.3679	0.8517	0.3174
SAN+LapPE [48]	0.6384	0.2683	0.1355	0.4004	0.8478	0.3350
SAN+RWSE [48]	0.6439	0.2545	0.1312	0.4030	0.8550	0.3341
GraphGPS [71]	0.6535	0.2500	-	-	-	0.3337
Exphormer [79]	0.6527	**0.2481**	-	-	-	0.3637
ViT-PS (Ours)	**0.7594**	0.2798	**0.2516**	**0.5518**	**0.8962**	**0.4504**

3.4 Real-World Graph Learning with Vision Transformer

Having observed that pre-trained ViT can learn graph tasks well when symmetrized with our method, we now provide a preliminary test of it in real-world graph learning. We use three real-world graph datasets from [28] that involve chemical and biological graphs. PCQM-Contact dataset contains 529,434 molecular graphs with 53 nodes in maximum, and the task is contact map prediction framed as link prediction (S_n equivariant), on whether two atoms would be proximal when the molecule is in 3D space. Peptides-func and Peptides-struct are based on the same set of 15,535 protein graphs with 444 nodes in maximum and the tasks are property prediction (S_n invariant), requiring multi-label classification for Peptides-func and regression for Peptides-struct. The tasks require complex understanding of how the amino acids of the proteins would interact in 3D space. We implement our method using a ViT-Base pre-trained on ImageNet-21k as the base model f_θ and a 3-layer GIN as the equivariant distribution $p_\omega(g|\mathbf{x})$, following our best model in Section 3.3. We use 10 samples for both training and testing. Further details can be found in Appendix A.3.6.

The results are in Table 4. In Peptides-func and PCQM-Contact, the pre-trained ViT symmetrized with our method achieves a significantly higher performance compared to both GNNs and graph transformers, improving previous best by a large margin[5] (0.6527 $\rightarrow$ 0.7594 for Peptides-func AP, 0.1355 $\rightarrow$ 0.2516 for PCQM-Contact Hits@1). This demonstrates the scalability of our method as Peptides-func involves 444 maximum nodes, and also its generality as it performs well for both S_n invariant (Peptides-func) and equivariant (PCQM-Contact) tasks. We also note that, unlike some baselines, our method does not require costly Laplacian eigenvectors to compute positional encoding. On the Peptides-struct, our method achieves a slightly lower performance to SOTA graph transformers while still better than GNNs. We conjecture that regression is harder for the model to learn due to its stochasticity in predictions, and leave improving regression performance as future work.

4 Conclusion

We presented probabilistic symmetrization, a general framework that learns a distribution of group transformations conditioned on input data for symmetrization of an arbitrary function. By characterizing that the only condition for such distribution is equivariance to data symmetry, we instantiated models for a wide range of groups, including symmetric, orthogonal, Euclidean groups and their product combinations. Our extensive experiments demonstrated that the proposed framework achieves consistent improvement over other symmetrization methods, and competitive or outperforms equivariant networks over various datasets. We also showed for the first time in literature that transferring pre-trained parameters across data in different symmetries can sometimes be surprisingly beneficial.

Acknowledgements This work was supported in part by the National Research Foundation of Korea (NRF2021R1C1C1012540 and NRF2021R1A4A3032834) and IITP grant (2021-0-00537, 2022-0-00926, 2022-0-00959 and 2021-0-02068) funded by the Korea government (MSIT).

[5]We note that the baseline architectures are constructed within 500k parameter budget as a convention [28], while we use an identical architecture to ViT-Base to leverage pre-trained representations.

References

[1] R. Abboud, İ. İ. Ceylan, M. Grohe, and T. Lukasiewicz. The surprising power of graph neural networks with random node initialization. In *IJCAI*, 2021. (Cited on 6, 22)

[2] M. Balcilar, P. Héroux, B. Gaüzère, P. Vasseur, S. Adam, and P. Honeine. Breaking the limits of message passing graph neural networks. In *ICML*, 2021. (Cited on 6, 7, 22)

[3] H. Bao, L. Dong, S. Piao, and F. Wei. Beit: BERT pre-training of image transformers. In *ICLR*, 2022. (Cited on 2)

[4] S. Basu, P. Katdare, P. Sattigeri, V. Chenthamarakshan, K. Driggs-Campbell, P. Das, and L. R. Varshney. Equivariant few-shot learning from pretrained models. *arXiv*, 2023. (Cited on 22)

[5] S. Basu, P. Sattigeri, K. N. Ramamurthy, V. Chenthamarakshan, K. R. Varshney, L. R. Varshney, and P. Das. Equi-tuning: Group equivariant fine-tuning of pretrained models. *arXiv*, 2022. (Cited on 2, 6, 22)

[6] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. F. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, Ç. Gülçehre, H. F. Song, A. J. Ballard, J. Gilmer, G. E. Dahl, A. Vaswani, K. R. Allen, C. Nash, V. Langston, C. Dyer, N. Heess, D. Wierstra, P. Kohli, M. M. Botvinick, O. Vinyals, Y. Li, and R. Pascanu. Relational inductive biases, deep learning, and graph networks. *arXiv*, 2018. (Cited on 1, 21)

[7] Y. Bengio, N. Léonard, and A. C. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv*, abs/1308.3432, 2013. (Cited on 4)

[8] B. Bevilacqua, F. Frasca, D. Lim, B. Srinivasan, C. Cai, G. Balamurugan, M. M. Bronstein, and H. Maron. Equivariant subgraph aggregation networks. In *ICLR*, 2022. (Cited on 21)

[9] B. Bloem-Reddy and Y. W. Teh. Probabilistic symmetries and invariant neural networks. *J. Mach. Learn. Res.*, 2020. (Cited on 2)

[10] A. Bogatskiy, S. Ganguly, T. Kipf, R. Kondor, D. W. Miller, D. Murnane, J. T. Offermann, M. Pettee, P. Shanahan, C. Shimmin, and S. Thais. Symmetry group equivariant architectures for physics. *arXiv*, 2022. (Cited on 1, 21)

[11] X. Bresson and T. Laurent. Residual gated graph convnets. *arXiv*, 2019. (Cited on 9, 10)

[12] M. M. Bronstein, J. Bruna, T. Cohen, and P. Velickovic. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv*, 2021. (Cited on 1, 21)

[13] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In *NeurIPS*, 2020. (Cited on 22)

[14] C. Cai and Y. Wang. A note on over-smoothing for graph neural networks. *arXiv*, 2020. (Cited on 4, 21)

[15] M. Chen, Z. Wei, Z. Huang, B. Ding, and Y. Li. Simple and deep graph convolutional networks. In *ICML*, 2020. (Cited on 10)

[16] Z. Chen, S. Villar, L. Chen, and J. Bruna. On the equivalence between graph isomorphism testing and function approximation with gnns. In *NeurIPS*, 2019. (Cited on 9)

[17] E. Chien, C. Pan, J. Peng, and O. Milenkovic. You are allset: A multiset function framework for hypergraph neural networks. In *ICLR*, 2022. (Cited on 22)

[18] T. Cohen and M. Welling. Group equivariant convolutional networks. In *ICML*, 2016. (Cited on 21)

[19] T. S. Cohen and M. Welling. Steerable cnns. In *ICLR*, 2017. (Cited on 21)

[20] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Math. Control. Signals Syst.*, 1989. (Cited on 3, 22)

[21] G. Dasoulas, L. D. Santos, K. Scaman, and A. Virmaux. Coloring graph neural networks for node disambiguation. In *IJCAI*, 2020. (Cited on 22)

[22] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NeurIPS*, 2016. (Cited on 7)

[23] C. Deng, O. Litany, Y. Duan, A. Poulenard, A. Tagliasacchi, and L. J. Guibas. Vector neurons: A general framework for so(3)-equivariant networks. In *ICCV*, 2021. (Cited on 1, 5, 8, 21, 23)

[24] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019. (Cited on 5, 22)

[25] T. Dinh, Y. Zeng, R. Zhang, Z. Lin, M. Gira, S. Rajput, J. Sohn, D. S. Papailiopoulos, and K. Lee. LIFT: language-interfaced fine-tuning for non-language machine learning tasks. In *NeurIPS*, 2022. (Cited on 22)

[26] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. (Cited on 2, 5, 6, 8, 9, 24, 25)

[27] V. P. Dwivedi, C. K. Joshi, T. Laurent, Y. Bengio, and X. Bresson. Benchmarking graph neural networks. *arXiv*, 2020. (Cited on 8, 9, 22, 25)

[28] V. P. Dwivedi, L. Rampásek, M. Galkin, A. Parviz, G. Wolf, A. T. Luu, and D. Beaini. Long range graph benchmark. In *NeurIPS*, 2022. (Cited on 10, 22)

[29] W. Falcon. Pytorch lightning. `https://github.com/Lightning-AI/lightning`, 2019. (Cited on 25)

[30] M. Finzi, M. Welling, and A. G. Wilson. A practical method for constructing equivariant multilayer perceptrons for arbitrary matrix groups. In *ICML*, 2021. (Cited on 4, 19, 21)

[31] F. Fuchs, D. E. Worrall, V. Fischer, and M. Welling. Se(3)-transformers: 3d roto-translation equivariant attention networks. In *NeurIPS*, 2020. (Cited on 8, 22)

[32] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. In *ICML*, 2017. (Cited on 22)

[33] D. Hendrycks and K. Gimpel. Gaussian error linear units (gelus). *arXiv*, 2018. (Cited on 24)

[34] K. Hornik, M. B. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 1989. (Cited on 1, 3, 22)

[35] W. Hu, B. Liu, J. Gomes, M. Zitnik, P. Liang, V. S. Pande, and J. Leskovec. Strategies for pre-training graph neural networks. In *ICLR*, 2020. (Cited on 10)

[36] A. Jaegle, S. Borgeaud, J. Alayrac, C. Doersch, C. Ionescu, D. Ding, S. Koppula, D. Zoran, A. Brock, E. Shelhamer, O. J. Hénaff, M. M. Botvinick, A. Zisserman, O. Vinyals, and J. Carreira. Perceiver IO: A general architecture for structured inputs & outputs. In *ICLR*, 2022. (Cited on 22)

[37] A. Jaegle, F. Gimeno, A. Brock, O. Vinyals, A. Zisserman, and J. Carreira. Perceiver: General perception with iterative attention. In *ICML*, 2021. (Cited on 22)

[38] C. K. Joshi, C. Bodnar, S. V. Mathis, T. Cohen, and P. Liò. On the expressive power of geometric graph neural networks. *arXiv*, 2023. (Cited on 1, 4, 21)

[39] S. Kaba, A. K. Mondal, Y. Zhang, Y. Bengio, and S. Ravanbakhsh. Equivariance with learned canonicalization functions. *arXiv*, 2022. (Cited on 2, 4, 5, 6, 7, 8, 15, 17, 18, 19, 21, 22)

[40] P. Kicki, P. Skrzypczynski, and M. Ozay. A new approach to design symmetry invariant neural networks. In *IJCNN*, 2021. (Cited on 2, 22)

[41] J. Kim, D. Nguyen, S. Min, S. Cho, M. Lee, H. Lee, and S. Hong. Pure transformers are powerful graph learners. In *NeurIPS*, 2022. (Cited on 22)

[42] J. Kim, S. Oh, S. Cho, and S. Hong. Equivariant hypergraph neural networks. In *ECCV*, 2022. (Cited on 22)

[43] J. Kim, S. Oh, and S. Hong. Transformers generalize deepsets and can be extended to graphs and hypergraphs. In *NeurIPS*, 2021. (Cited on 1, 22)

[44] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. (Cited on 24, 25)

[45] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *ICLR*, 2014. (Cited on 2)

[46] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017. (Cited on 7, 9, 10)

[47] J. Köhler, L. Klein, and F. Noé. Equivariant flows: Sampling configurations for multi-body systems with symmetric energies. *arXiv*, 2019. (Cited on 8)

[48] D. Kreuzer, D. Beaini, W. L. Hamilton, V. Létourneau, and P. Tossou. Rethinking graph transformers with spectral attention. *NeurIPS*, 2021. (Cited on 10, 22)

[49] J. Lee, Y. Lee, J. Kim, A. R. Kosiorek, S. Choi, and Y. W. Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *ICML*, 2019. (Cited on 22)

[50] J. Li, D. Li, S. Savarese, and S. C. H. Hoi. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv*, 2023. (Cited on 22)

[51] W. Liu, A. Rabinovich, and A. C. Berg. Parsenet: Looking wider to see better. *arXiv*, 2015. (Cited on 25)

[52] K. Lu, A. Grover, P. Abbeel, and I. Mordatch. Frozen pretrained transformers as universal computation engines. In *AAAI*, 2022. (Cited on 22)

[53] S. Luo, T. Chen, Y. Xu, S. Zheng, T. Liu, D. He, and L. Wang. One transformer can understand both 2d & 3d molecular data. *arXiv*, 2022. (Cited on 22)

[54] H. Maron, H. Ben-Hamu, H. Serviansky, and Y. Lipman. Provably powerful graph networks. In *NeurIPS*, 2019. (Cited on 1, 6, 7, 9, 21)

[55] H. Maron, H. Ben-Hamu, N. Shamir, and Y. Lipman. Invariant and equivariant graph networks. In *ICLR*, 2019. (Cited on 19, 21)

[56] H. Maron, E. Fetaya, N. Segol, and Y. Lipman. On the universality of invariant networks. In *ICML*, 2019. (Cited on 1)

[57] H. Maron, O. Litany, G. Chechik, and E. Fetaya. On learning sets of symmetric elements. In *ICML*, 2020. (Cited on 19)

[58] F. Mezzadri. How to generate random matrices from the classical compact groups. *Notices of the American Mathematical Society*, 2006. (Cited on 2)

[59] E. Min, R. Chen, Y. Bian, T. Xu, K. Zhao, W. Huang, P. Zhao, J. Huang, S. Ananiadou, and Y. Rong. Transformer for graphs: An overview from architecture perspective. *arXiv*, 2022. (Cited on 22)

[60] C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, and M. Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *AAAI*, 2019. (Cited on 1, 21)

[61] S. C. Mouli and B. Ribeiro. Neural networks for learning counterfactual g-invariances from single environments. In *ICLR*, 2021. (Cited on 2, 22)

[62] L. Müller, M. Galkin, C. Morris, and L. Rampásek. Attending to graph transformers. *arXiv*, 2023. (Cited on 22)

[63] R. L. Murphy, B. Srinivasan, V. Rao, and B. Ribeiro. Janossy pooling: Learning deep permutation-invariant functions for variable-size inputs. In *ICLR*, 2019. (Cited on 1, 2, 22)

[64] R. L. Murphy, B. Srinivasan, V. A. Rao, and B. Ribeiro. Relational pooling for graph representations. In *ICML*, 2019. (Cited on 2, 22)

[65] H. NT and T. Maehara. Revisiting graph neural networks: All we have is low-pass filters. *arXiv*, 2019. (Cited on 21)

[66] K. Oono and T. Suzuki. Graph neural networks exponentially lose expressive power for node classification. In *ICLR*, 2020. (Cited on 4, 21)

[67] F. Paischer, T. Adler, V. P. Patil, A. Bitto-Nemling, M. Holzleitner, S. Lehner, H. Eghbal-Zadeh, and S. Hochreiter. History compression via language models in reinforcement learning. In *ICML*, 2022. (Cited on 22)

[68] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. (Cited on 24)

[69] O. Puny, M. Atzmon, E. J. Smith, I. Misra, A. Grover, H. Ben-Hamu, and Y. Lipman. Frame averaging for invariant and equivariant network design. In *ICLR*, 2022. (Cited on 2, 5, 6, 7, 8, 15, 18, 19, 21, 22, 24)

[70] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 2020. (Cited on 22)

[71] L. Rampásek, M. Galkin, V. P. Dwivedi, A. T. Luu, G. Wolf, and D. Beaini. Recipe for a general, powerful, scalable graph transformer. In *NeurIPS*, 2022. (Cited on 10)

[72] S. Ravanbakhsh, J. G. Schneider, and B. Póczos. Equivariance through parameter-sharing. In *ICML*, 2017. (Cited on 21)

[73] L. F. R. Ribeiro, M. Schmitt, H. Schütze, and I. Gurevych. Investigating pretrained language models for graph-to-text generation. *arXiv*, 2020. (Cited on 22)

[74] D. Rothermel, M. Li, T. Rocktäschel, and J. Foerster. Don't sweep your learning rate under the rug: A closer look at cross-modal transfer of pretrained transformers. *arXiv*, 2021. (Cited on 22)

[75] A. Sannai, M. Kawano, and W. Kumagai. Equivariant and invariant reynolds networks. *arXiv*, 2021. (Cited on 2)

[76] V. G. Satorras, E. Hoogeboom, and M. Welling. E(n) equivariant graph neural networks. In *ICML*, 2021. (Cited on 8, 21, 22)

[77] J. Shen, L. Li, L. M. Dery, C. Staten, M. Khodak, G. Neubig, and A. Talwalkar. Cross-modal fine-tuning: Align then refine. *arXiv*, 2023. (Cited on 5, 22)

[78] W. Shi, J. Caballero, F. Huszar, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, 2016. (Cited on 5)

[79] H. Shirzad, A. Velingker, B. Venkatachalam, D. J. Sutherland, and A. K. Sinop. Exphormer: Sparse transformers for graphs. *arXiv*, 2023. (Cited on 10)

[80] B. Srinivasan and B. Ribeiro. On the equivalence between positional node embeddings and structural graph representations. In *ICLR*, 2020. (Cited on 22)

[81] N. Thomas, T. E. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley. Tensor field networks: Rotation- and translation-equivariant neural networks for 3d point clouds. *arXiv*, 2018. (Cited on 1, 8, 21)

[82] J. Topping, F. D. Giovanni, B. P. Chamberlain, X. Dong, and M. M. Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. In *ICLR*, 2022. (Cited on 21)

[83] E. van der Pol, D. E. Worrall, H. van Hoof, F. A. Oliehoek, and M. Welling. MDP homomorphic networks: Group symmetries in reinforcement learning. In *NeurIPS*, 2020. (Cited on 2, 22)

[84] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *NeurIPS*, 2017. (Cited on 22, 24)

[85] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks. In *ICLR*, 2018. (Cited on 7, 9)

[86] R. Winter, F. Noé, and D. Clevert. Permutation-invariant variational autoencoder for graph-level representation learning. In *NeurIPS*, 2021. (Cited on 4, 17, 23)

[87] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, and J. Brew. Huggingface's transformers: State-of-the-art natural language processing. *arXiv*, 2019. (Cited on 6, 24)

[88] R. Xiong, Y. Yang, D. He, K. Zheng, S. Zheng, C. Xing, H. Zhang, Y. Lan, L. Wang, and T. Liu. On layer normalization in the transformer architecture. In *ICML*, 2020. (Cited on 24)

[89] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? In *ICLR*, 2019. (Cited on 1, 4, 6, 7, 9, 21, 22)

[90] D. Yarotsky. Universal approximations of invariant maps by neural networks. *arXiv*, 2018. (Cited on 1, 2, 6, 7, 15, 22)

[91] C. Ying, T. Cai, S. Luo, S. Zheng, G. Ke, D. He, Y. Shen, and T. Liu. Do transformers really perform bad for graph representation? In *NeurIPS*, 2021. (Cited on 22)

[92] C. Yun, S. Bhojanapalli, A. S. Rawat, S. J. Reddi, and S. Kumar. Are transformers universal approximators of sequence-to-sequence functions? In *ICLR*, 2020. (Cited on 1, 3, 6, 22)

[93] B. Zhang, S. Luo, L. Wang, and D. He. Rethinking the expressive power of gnns via graph biconnectivity. *arXiv*, 2023. (Cited on 1, 21)

A Appendix

A.1 Proofs

A.1.1 Proof of Theorem 1 (Section 2.1)

Theorem 1. *If p_ω is G equivariant, then $\phi_{\theta,\omega}$ is G equivariant for arbitrary f_θ.*

Proof. We prove $\phi_{\theta,\omega}(\rho_1(g')\mathbf{x}) = \rho_2(g')\phi_{\theta,\omega}(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{X}$ and $g' \in G$. From Eq. (4), we have:

$$\phi_{\theta,\omega}(\rho_1(g')\mathbf{x}) = \mathbb{E}_{p_\omega(g|\rho_1(g')\mathbf{x})}\left[\rho_2(g) f_\theta(\rho_1(g)^{-1}\rho_1(g')\mathbf{x})\right]. \tag{11}$$

Let us introduce transformed random variable $h = g'^{-1}g \in G$ such that $g = g'h$. Since the distribution p_ω is G equivariant, we can see that $p_\omega(g|\rho_1(g')\mathbf{x}) = p_\omega(g'^{-1}g|\rho_1(g'^{-1})\rho_1(g')\mathbf{x}) = p_\omega(g'^{-1}g|\mathbf{x}) = p_\omega(h|\mathbf{x})$. Thus, we can rewrite the above expectation with respect to h as follows:

$$\begin{aligned}
\phi_{\theta,\omega}(\rho_1(g')\mathbf{x}) &= \mathbb{E}_{p_\omega(h|\mathbf{x})}\left[\rho_2(g'h) f_\theta(\rho_1(g'h)^{-1}\rho_1(g')\mathbf{x})\right] \\
&= \mathbb{E}_{p_\omega(h|\mathbf{x})}\left[\rho_2(g')\rho_2(h) f_\theta(\rho_1(h)^{-1}\rho_1(g')^{-1}\rho_1(g')\mathbf{x})\right] \\
&= \rho_2(g')\mathbb{E}_{p_\omega(h|\mathbf{x})}\left[\rho_2(h) f_\theta(\rho_1(h)^{-1}\mathbf{x})\right] \\
&= \rho_2(g')\phi_{\theta,\omega}(\mathbf{x}),
\end{aligned} \tag{12}$$

showing the G equivariance of $\phi_{\theta,\omega}$ for arbitrary f_θ. □

A.1.2 Proof of Theorem 2 (Section 2.1)

Theorem 2. *If p_ω is G equivariant and f_θ is a universal approximator, then $\phi_{\theta,\omega}$ is a universal approximator of G equivariant functions.*

Proof. The proof is inspired by universality proofs of prior symmetrization approaches [90, 69, 39]. Let $\psi : \mathcal{X} \to \mathcal{Y}$ be an arbitrary G equivariant function. By equivariance of ψ, we have:

$$\begin{aligned}
\|\psi(\mathbf{x}) - \phi_{\theta,\omega}(\mathbf{x})\| &= \left\|\psi(\mathbf{x}) - \mathbb{E}_{p_\omega(g|\mathbf{x})}\left[\rho_2(g) f_\theta(\rho_1(g)^{-1}\mathbf{x})\right]\right\| \\
&= \left\|\mathbb{E}_{p_\omega(g|\mathbf{x})}\left[\psi(\mathbf{x})\right] - \mathbb{E}_{p_\omega(g|\mathbf{x})}\left[\rho_2(g) f_\theta(\rho_1(g)^{-1}\mathbf{x})\right]\right\| \\
&= \left\|\mathbb{E}_{p_\omega(g|\mathbf{x})}\left[\rho_2(g)\rho_2(g)^{-1}\psi(\mathbf{x})\right] - \mathbb{E}_{p_\omega(g|\mathbf{x})}\left[\rho_2(g) f_\theta(\rho_1(g)^{-1}\mathbf{x})\right]\right\| \\
&= \left\|\mathbb{E}_{p_\omega(g|\mathbf{x})}\left[\rho_2(g)\psi(\rho_1(g)^{-1}\mathbf{x})\right] - \mathbb{E}_{p_\omega(g|\mathbf{x})}\left[\rho_2(g) f_\theta(\rho_1(g)^{-1}\mathbf{x})\right]\right\| \\
&= \left\|\mathbb{E}_{p_\omega(g|\mathbf{x})}\left[\rho_2(g)\psi(\rho_1(g)^{-1}\mathbf{x}) - \rho_2(g) f_\theta(\rho_1(g)^{-1}\mathbf{x})\right]\right\|.
\end{aligned} \tag{13}$$

As $\mathcal{Y}$ is finite-dimensional, we can assume that the linear operators in $\mathrm{GL}(\mathcal{Y})$ are bounded and so is the induced operator norm of group representation $\|\rho_2(g)\|$ for all $g \in G$. Thus, we have:

$$\begin{aligned}
\|\psi(\mathbf{x}) - \phi_{\theta,\omega}(\mathbf{x})\| &\le \max_{h\in G}\|\rho_2(h)\|\left\|\mathbb{E}_{p_\omega(g|\mathbf{x})}\left[\psi(\rho_1(g)^{-1}\mathbf{x}) - f_\theta(\rho_1(g)^{-1}\mathbf{x})\right]\right\| \\
&\le c\left\|\mathbb{E}_{p_\omega(g|\mathbf{x})}\left[\psi(\rho_1(g)^{-1}\mathbf{x}) - f_\theta(\rho_1(g)^{-1}\mathbf{x})\right]\right\|.
\end{aligned} \tag{14}$$

for some $c > 0$. If f_θ is a universal approximator, for any compact set $\mathcal{K} \subseteq \mathcal{X}$ and any $\epsilon > 0$, there exists some θ such that $\|\psi(\mathbf{x}) - f_\theta(\mathbf{x})\| \le \epsilon$ for all $\mathbf{x} \in \mathcal{K}$. Consider the set $\mathcal{K}_{\text{sym}} = \cup_{g\in G}\rho_1(g)\mathcal{K}$ where $\rho_1(g)\mathcal{K}$ denotes the image of the set $\mathcal{K}$ under linear transformation by $\rho_1(g)$. We use the fact that $\mathcal{K}_{\text{sym}}$ is also a compact set since it is the image of the compact set $G \times \mathcal{K}$ under continuous map $(g, \mathbf{x}) \mapsto \rho_1(g)\mathbf{x}$. As a consequence, for any compact set $\mathcal{K} \subseteq \mathcal{X}$ and any $\epsilon/c > 0$, there exists some θ such that $\max_{g\in G}\|\psi(\rho_1(g)\mathbf{x}) - f_\theta(\rho_1(g)\mathbf{x})\| \le \epsilon/c$ for all $\mathbf{x} \in \mathcal{K}$. Since a group is closed under inverse, for any compact set $\mathcal{K} \subseteq \mathcal{X}$ and any $\epsilon > 0$, there exists some θ such that:

$$\begin{aligned}
\|\psi(\mathbf{x}) - \phi_{\theta,\omega}(\mathbf{x})\| &\le c\left\|\mathbb{E}_{p_\omega(g|\mathbf{x})}\left[\psi(\rho_1(g)^{-1}\mathbf{x}) - f_\theta(\rho_1(g)^{-1}\mathbf{x})\right]\right\| \\
&\le c\max_{g\in G}\left\|\psi(\rho_1(g)^{-1}\mathbf{x}) - f_\theta(\rho_1(g)^{-1}\mathbf{x})\right\| \\
&= \epsilon,
\end{aligned} \tag{15}$$

for all $\mathbf{x} \in \mathcal{K}$, showing that $\phi_{\theta,\omega}$ is a universal approximator of G equivariant functions. □

A.1.3 Proof of Theorem 3 (Section 2.1)

Theorem 3. *If q_ω is G equivariant and $p(\boldsymbol{\epsilon})$ is G invariant under a faithful orthogonal group representation ρ', then the distribution $p_\omega(g|\mathbf{x})$ characterized by $q_\omega : (\mathbf{x}, \boldsymbol{\epsilon}) \mapsto \rho(g)$ is G equivariant.*

Proof. We prove $p_\omega(g'g|\rho_1(g')\mathbf{x}) = p_\omega(g|\mathbf{x})$ for all $\mathbf{x} \in \mathcal{X}$ and $g, g' \in G$. In general, we are interested in obtaining a faithful representation ρ, *i.e.*, such that $\rho(g)$ is distinct for each g. We can interpret the probability $p_\omega(g|\mathbf{x}, \boldsymbol{\epsilon})$ as a delta distribution centered at the group representation $\rho(g)$:

$$p_\omega(g|\mathbf{x}, \boldsymbol{\epsilon}) = \delta(\rho(g) = q_\omega(\mathbf{x}, \boldsymbol{\epsilon})). \tag{16}$$

To obtain $p_\omega(g|\mathbf{x})$, we marginalize over $p(\boldsymbol{\epsilon})$:

$$\begin{aligned} p_\omega(g|\mathbf{x}) &= \int_{\boldsymbol{\epsilon}} p_\omega(g|\mathbf{x}, \boldsymbol{\epsilon}) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon} \\ &= \int_{\boldsymbol{\epsilon}} \delta(\rho(g) = q_\omega(\mathbf{x}, \boldsymbol{\epsilon})) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon}. \end{aligned} \tag{17}$$

Let us consider $p_\omega(g'g|\rho_1(g')\mathbf{x})$:

$$p_\omega(g'g|\rho_1(g')\mathbf{x}) = \int_{\boldsymbol{\epsilon}} \delta(\rho(g'g) = q_\omega(\rho_1(g')\mathbf{x}, \boldsymbol{\epsilon})) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon}. \tag{18}$$

Using the G equivariance of q_ω, we have:

$$\begin{aligned} q_\omega(\rho_1(g')\mathbf{x}, \boldsymbol{\epsilon}) &= \rho(g') q_\omega(\rho_1(g'^{-1})\rho_1(g')\mathbf{x}, \rho'(g'^{-1})\boldsymbol{\epsilon}) \\ &= \rho(g') q_\omega(\mathbf{x}, \rho'(g'^{-1})\boldsymbol{\epsilon}) \end{aligned} \tag{19}$$

which leads to the following:

$$\begin{aligned} p_\omega(g'g|\rho_1(g')\mathbf{x}) &= \int_{\boldsymbol{\epsilon}} \delta(\rho(g'g) = \rho(g') q_\omega(\mathbf{x}, \rho'(g'^{-1})\boldsymbol{\epsilon})) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon} \\ &= \int_{\boldsymbol{\epsilon}} \delta(\rho(g) = q_\omega(\mathbf{x}, \rho'(g'^{-1})\boldsymbol{\epsilon})) p(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon}. \end{aligned} \tag{20}$$

Note that the second equality follows from invertibility of $\rho(g')$. We now introduce a change of variables $\boldsymbol{\epsilon}' = \rho'(g'^{-1})\boldsymbol{\epsilon}$ that $\boldsymbol{\epsilon} = \rho'(g')\boldsymbol{\epsilon}'$:

$$p_\omega(g'g|\rho_1(g')\mathbf{x}) = \int_{\boldsymbol{\epsilon}'} \delta(\rho(g) = q_\omega(\mathbf{x}, \boldsymbol{\epsilon}')) p(\rho'(g')\boldsymbol{\epsilon}') \frac{1}{|\det \rho'(g'^{-1})|} d\boldsymbol{\epsilon}'. \tag{21}$$

With the fact that ρ' is an orthogonal representation which gives $|\det \rho'(g'^{-1})| = 1$, and G invariance of $p(\boldsymbol{\epsilon})$ which gives $p(\rho'(g')\boldsymbol{\epsilon}') = p(\boldsymbol{\epsilon}')$, we get:

$$\begin{aligned} p_\omega(g'g|\rho_1(g')\mathbf{x}) &= \int_{\boldsymbol{\epsilon}'} \delta(\rho(g) = q_\omega(\mathbf{x}, \boldsymbol{\epsilon}')) p(\boldsymbol{\epsilon}') d\boldsymbol{\epsilon}' \\ &= p_\omega(g|\mathbf{x}), \end{aligned} \tag{22}$$

showing the G equivariance of $p_\omega(g|\mathbf{x})$. □

A.1.4 Proof of Validity for Implemented Equivariant Distributions p_ω (Section 2.2)

We formally show G equivariance of the implemented distributions $p_\omega(g|\mathbf{x})$ presented in Section 2.2. All implementations have a form of noise-outsourced function $q_\omega : (\mathbf{x}, \boldsymbol{\epsilon}) \mapsto \rho(g)$ using distribution $\boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon})$ and map q_ω which is composed of G equivariant neural network and postprocessing to $\rho(g)$. From Theorem 3, for G equivariance of $p_\omega(g|\mathbf{x})$, it is sufficient to show G invariance of $p(\boldsymbol{\epsilon})$ under a faithful orthogonal representation ρ' along with G equivariance of q_ω, which we show below.

Symmetric Group S_n We recall that $p_\omega(g|\mathbf{x})$ for the symmetric group S_n is implemented as below:

1. Sample node-level noise $\boldsymbol{\epsilon} \in \mathbb{R}^{n \times d}$ from i.i.d. uniform $\mathrm{Unif}[0, \eta]$.
2. Use a GNN to obtain node-level scalar features $(\mathbf{x}, \boldsymbol{\epsilon}) \mapsto \mathbf{Z} \in \mathbb{R}^n$.

3. Assuming $\mathbf{Z}$ is tie-free, use below argsort [86] to obtain group representation $\mathbf{Z} \mapsto \mathbf{P}_g = \rho(g)$.

$$\mathbf{P}_g \approx \hat{\mathbf{P}}_g = \text{softmax}\left(\frac{-\|\mathbf{Z}\mathbf{1}^\top - \mathbf{1}\text{sort}(\mathbf{Z})^\top\|_1}{\tau}\right), \tag{23}$$

where softmax applies to each row and $\tau \in \mathbb{R}_+$ is a temperature hyperparameter.

We now show the following:

Proposition 3. *The proposed distribution $p_\omega(g|\mathbf{x})$ for the symmetric group S_n is equivariant.*

Proof. Since $p(\boldsymbol{\epsilon})$ is an elementwise i.i.d., it is S_n invariant under the base representation $\rho'(g) = \mathbf{P}_g$ which is faithful and orthogonal. As a GNN is S_n equivariant, we only need to show S_n equivariance of argsort : $\mathbf{Z} \mapsto \mathbf{P}_g$. This can be shown by transforming $\mathbf{Z}$ with arbitrary permutation matrix $\mathbf{P}_{g'}$. First, as sort operator and any row replicated matrix are invariant to $\mathbf{P}_{g'}$, we have the following:

$$\begin{aligned}\text{softmax}\left(\frac{-\|\mathbf{P}_{g'}\mathbf{Z}\mathbf{1}^\top - \mathbf{1}\text{sort}(\mathbf{P}_{g'}\mathbf{Z})^\top\|_1}{\tau}\right) &= \text{softmax}\left(\frac{-\|\mathbf{P}_{g'}\mathbf{Z}\mathbf{1}^\top - \mathbf{1}\text{sort}(\mathbf{Z})^\top\|_1}{\tau}\right) \\ &= \text{softmax}\left(\frac{-\|\mathbf{P}_{g'}\mathbf{Z}\mathbf{1}^\top - \mathbf{P}_{g'}\mathbf{1}\text{sort}(\mathbf{Z})^\top\|_1}{\tau}\right).\end{aligned} \tag{24}$$

Since L1 norm, scaling with $-1/\tau$, and row-wise softmax all commute with $\mathbf{P}_{g'}$, we have:

$$\begin{aligned}\text{softmax}\left(\frac{-\|\mathbf{P}_{g'}\mathbf{Z}\mathbf{1}^\top - \mathbf{1}\text{sort}(\mathbf{P}_{g'}\mathbf{Z})^\top\|_1}{\tau}\right) &= \text{softmax}\left(\mathbf{P}_{g'}\frac{-\|\mathbf{Z}\mathbf{1}^\top - \mathbf{1}\text{sort}(\mathbf{Z})^\top\|_1}{\tau}\right) \\ &= \mathbf{P}_{g'}\text{softmax}\left(\frac{-\|\mathbf{Z}\mathbf{1}^\top - \mathbf{1}\text{sort}(\mathbf{Z})^\top\|_1}{\tau}\right) \\ &= \mathbf{P}_{g'}\hat{\mathbf{P}}_g \approx \mathbf{P}_{g'}\mathbf{P}_g,\end{aligned} \tag{25}$$

showing that argsort is S_n equivariant, *i.e.*, it maps $\mathbf{P}_{g'}\mathbf{Z} \mapsto \mathbf{P}_{g'}\mathbf{P}_g$ for all $\mathbf{P}_{g'} \in \mathrm{S}_n$. Combining the above, by Theorem 3, the distribution $p_\omega(g|\mathbf{x})$ is S_n equivariant. □

Orthogonal Group $\mathrm{O}(n)$, $\mathrm{SO}(n)$ We recall that $p_\omega(g|\mathbf{x})$ for the orthogonal group $\mathrm{O}(n)$ or special orthogonal group $\mathrm{SO}(n)$ is implemented as follows:

1. Sample noise $\boldsymbol{\epsilon} \in \mathbb{R}^{n\times d}$ from i.i.d. normal $\mathcal{N}(0, \eta^2)$.
2. Use an $\mathrm{O}(n)$/$\mathrm{SO}(n)$ equivariant neural network to obtain n features $(\mathbf{x}, \boldsymbol{\epsilon}) \mapsto \mathbf{Z} \in \mathbb{R}^{n\times n}$.
3. Assuming $\mathbf{Z}$ is full-rank, use Gram-Schmidt process [39] to obtain an orthogonal matrix $\mathbf{Z} \mapsto \mathbf{Q}$.
4. For the $\mathrm{O}(n)$ group, use the obtained matrix as group representation $\mathbf{Q} = \mathbf{Q}_g = \rho(g)$.
5. For the $\mathrm{SO}(n)$ group, use below scale operator to obtain group representation $\mathbf{Q} \mapsto \mathbf{Q}_g^+ = \rho(g)$.

$$\text{scale} : \left[\begin{array}{c|c|c} \mathbf{Q}_1 & \ldots & \mathbf{Q}_n \end{array}\right] \mapsto \left[\begin{array}{c|c|c} \det(\mathbf{Q})\cdot\mathbf{Q}_1 & \ldots & \mathbf{Q}_n \end{array}\right]. \tag{26}$$

We now show the following:

Proposition 4. *The proposed distribution $p_\omega(g|\mathbf{x})$ for the orthogonal group $\mathrm{O}(n)$ is equivariant.*

Proof. Without loss of generality, let us omit the scale η for brevity, which gives that each column $\boldsymbol{\epsilon}_i \in \mathbb{R}^n$ of the noise $\boldsymbol{\epsilon}$ independently follows multivariate standard normal $\boldsymbol{\epsilon}_i \sim \mathcal{N}(0, \mathbf{I}_n)$. Then, the density $p(\boldsymbol{\epsilon}_i) = (2\pi)^{-n/2}\exp\left(-\|\boldsymbol{\epsilon}_i\|_2^2/2\right)$ is invariant under orthogonal transformation $\mathbf{Q}$ since $\|\mathbf{Q}\boldsymbol{\epsilon}_i\|_2^2 = (\mathbf{Q}\boldsymbol{\epsilon}_i)^\top\mathbf{Q}\boldsymbol{\epsilon}_i = \boldsymbol{\epsilon}_i^\top\mathbf{Q}^\top\mathbf{Q}\boldsymbol{\epsilon}_i = \boldsymbol{\epsilon}_i^\top\boldsymbol{\epsilon}_i = \|\boldsymbol{\epsilon}_i\|_2^2$. Therefore, the distribution $p(\boldsymbol{\epsilon})$ is invariant under the base representation $\rho'(g) = \mathbf{Q}_g$, which is faithful and orthogonal. As we use an equivariant neural network to obtain $\mathbf{Z}$, and Gram-Schmidt procedure that maps $\mathbf{Z} \mapsto \mathbf{Q}_g$ is $\mathrm{O}(n)$ equivariant (Theorem 5 of [39]), by Theorem 3, the distribution $p_\omega(g|\mathbf{x})$ is $\mathrm{O}(n)$ equivariant. □

Proposition 5. *The proposed distribution $p_\omega(g|\mathbf{x})$ for special orthogonal group $\mathrm{SO}(n)$ is equivariant.*

Proof. From the proof of Proposition 4, it follows that the distribution $p(\boldsymbol{\epsilon})$ is invariant under the base representation $\rho'(g) = \mathbf{Q}_g^+$ which is faithful and orthogonal. As we use an equivariant neural network to obtain $\mathbf{Z}$, and Gram-Schmidt procedure $\mathbf{Z} \mapsto \mathbf{Q}$ has $\mathrm{O}(n)$ equivariance which implies $\mathrm{SO}(n)$ equivariance because of $\mathrm{SO}(n) \leq \mathrm{O}(n)$, we only need to show $\mathrm{SO}(n)$ equivariance of scale : $\mathbf{Q} \mapsto \mathbf{Q}_g^+$. This can be done by transforming $\mathbf{Q}$ with an orthogonal $\mathbf{Q}_{g'}^+$ of determinant $+1$. Since $\det(\mathbf{Q}_{g'}^+\mathbf{Q}) = \det(\mathbf{Q}_{g'}^+)\det(\mathbf{Q}) = \det(\mathbf{Q})$, we have the following:

$$\begin{aligned}\mathrm{scale}(\mathbf{Q}_{g'}^+\mathbf{Q}) &= \left[\begin{array}{c|c|c} \det(\mathbf{Q}_{g'}^+\mathbf{Q}) \cdot (\mathbf{Q}_{g'}^+\mathbf{Q})_1 & \ldots & (\mathbf{Q}_{g'}^+\mathbf{Q})_n \end{array}\right] \\ &= \left[\begin{array}{c|c|c} \det(\mathbf{Q}) \cdot (\mathbf{Q}_{g'}^+\mathbf{Q})_1 & \ldots & (\mathbf{Q}_{g'}^+\mathbf{Q})_n \end{array}\right]. \end{aligned} \tag{27}$$

Also, scaling the first column of the product $\mathbf{Q}_{g'}^+\mathbf{Q}$ with $\det(\mathbf{Q})$ is equivalent to scaling the first column of $\mathbf{Q}$ with $\det(\mathbf{Q})$ then computing the product since $(\mathbf{Q}_{g'}^+\mathbf{Q})_{ij} = \sum_k \mathbf{Q}_{g'ik}^+\mathbf{Q}_{kj}$. This gives:

$$\begin{aligned}\mathrm{scale}(\mathbf{Q}_{g'}^+\mathbf{Q}) &= \mathbf{Q}_{g'}^+ \left[\begin{array}{c|c|c} \det(\mathbf{Q}) \cdot \mathbf{Q}_1 & \ldots & \mathbf{Q}_n \end{array}\right] \\ &= \mathbf{Q}_{g'}^+ \mathrm{scale}(\mathbf{Q}), \end{aligned} \tag{28}$$

showing that scale operator is $\mathrm{SO}(n)$ equivariant. We also note that $\mathrm{scale}(\mathbf{Q})$ gives orthogonal matrix of determinant $+1$, as it returns $\mathbf{Q}$ if $\det(\mathbf{Q}) = +1$, otherwise ($\det(\mathbf{Q}) = -1$ since $\mathbf{Q}$ is orthogonal) scales the first column by -1 which flips determinant to $+1$ while not affecting orthogonality. Combining the above, by Theorem 3, the distribution $p_\omega(g|\mathbf{x})$ is $\mathrm{SO}(n)$ equivariant. □

Euclidean Group $\mathrm{E}(n)$**,** $\mathrm{SE}(n)$ We recall that, unlike the other groups, we handle the Euclidean group $\mathrm{E}(n)$ and special Euclidean group $\mathrm{SE}(n)$ at symmetrization level as the translation component $\mathrm{T}(n)$ in $\mathrm{E}(n) = \mathrm{O}(n) \ltimes \mathrm{T}(n)$ and $\mathrm{SE}(n) = \mathrm{SO}(n) \ltimes \mathrm{T}(n)$ is non-compact. This is done as follows:

$$\phi_{\theta,\omega}(\mathbf{x}) = \mathbb{E}_{p_\omega(g|\mathbf{x}-\bar{\mathbf{x}}\mathbf{1}^\top)}\left[\bar{\mathbf{x}}\mathbf{1}^\top + g \cdot f_\theta(g^{-1} \cdot (\mathbf{x} - \bar{\mathbf{x}}\mathbf{1}^\top))\right], \tag{29}$$

where $\bar{\mathbf{x}} \in \mathbb{R}^n$ is centroid (mean over channels) of data $\mathbf{x} \in \mathbb{R}^{n\times d}$ and distribution p_ω is $\mathrm{O}(n)$/$\mathrm{SO}(n)$ equivariant for $\mathrm{E}(n)$/$\mathrm{SE}(n)$ equivariant symmetrization, respectively. We now show the following:

Proposition 6. *The proposed symmetrization $\phi_{\theta,\omega}$ for the Euclidean group $\mathrm{E}(n)$ is equivariant.*

Proof. We prove $\phi_{\theta,\omega}(g' \cdot \mathbf{x}) = g' \cdot \phi_{\theta,\omega}(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{X}$ and $g' \in \mathrm{E}(n)$. From Eq. (29), we have:

$$\phi_{\theta,\omega}(g' \cdot \mathbf{x}) = \mathbb{E}_{p_\omega(g|g'\cdot\mathbf{x}-\overline{g'\cdot\mathbf{x}}\mathbf{1}^\top)}\left[\,\overline{g'\cdot\mathbf{x}}\mathbf{1}^\top + g \cdot f_\theta(g^{-1} \cdot (g' \cdot \mathbf{x} - \overline{g'\cdot\mathbf{x}}\mathbf{1}^\top))\right]. \tag{30}$$

In general, an element of Euclidean group $g' \in \mathrm{E}(n)$ acts on data $\mathbf{x} \in \mathbb{R}^{n\times d}$ via $g'\cdot\mathbf{x} = \mathbf{Q}_{g'}\mathbf{x} + \mathbf{t}_{g'}\mathbf{1}^\top$ where $\mathbf{Q}_{g'} \in \mathrm{O}(n)$ is its rotation component and $\mathbf{t}_{g'} \in \mathbb{R}^n$ is its translation component [69, 39]. With this, the centroid of the transformed data $g' \cdot \mathbf{x}$ is given as follows:

$$\overline{g'\cdot\mathbf{x}} = \overline{\mathbf{Q}_{g'}\mathbf{x} + \mathbf{t}_{g'}\mathbf{1}^\top} = \overline{\mathbf{Q}_{g'}\mathbf{x}} + \mathbf{t}_{g'} = \mathbf{Q}_{g'}\overline{\mathbf{x}} + \mathbf{t}_{g'}, \tag{31}$$

which leads to the following:

$$\begin{aligned} g' \cdot \mathbf{x} - \overline{g'\cdot\mathbf{x}}\mathbf{1}^\top &= \mathbf{Q}_{g'}\mathbf{x} + \mathbf{t}_{g'}\mathbf{1}^\top - \mathbf{Q}_{g'}\overline{\mathbf{x}}\mathbf{1}^\top - \mathbf{t}_{g'}\mathbf{1}^\top \\ &= \mathbf{Q}_{g'}(\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top). \end{aligned} \tag{32}$$

Above shows that subtracting centroid eliminates the translation component of the problem and leaves $\mathrm{O}(n)$ equivariance component. Based on that, we have the following:

$$\begin{aligned}\phi_{\theta,\omega}(g' \cdot \mathbf{x}) &= \mathbb{E}_{p_\omega(g|\mathbf{Q}_{g'}(\mathbf{x}-\overline{\mathbf{x}}\mathbf{1}^\top))}\left[\mathbf{Q}_{g'}\bar{\mathbf{x}}\mathbf{1}^\top + \mathbf{t}_{g'}\mathbf{1}^\top + g \cdot f_\theta(g^{-1} \cdot (\mathbf{Q}_{g'}(\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top)))\right] \\ &= \mathbb{E}_{p_\omega(g|g'\cdot(\mathbf{x}-\overline{\mathbf{x}}\mathbf{1}^\top))}\left[g' \cdot \bar{\mathbf{x}}\mathbf{1}^\top + g \cdot f_\theta(g^{-1}g' \cdot (\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top))\right] + \mathbf{t}_{g'}\mathbf{1}^\top. \end{aligned} \tag{33}$$

Note that, inside the expectation, we interpret the rotation component of g' as an element of the orthogonal group $\mathrm{O}(n)$. Similar as in the proof of Theorem 1, we introduce transformed random

variable $h = g'^{-1}g \in \mathrm{O}(n)$ that $g = g'h$. Since the distribution p_ω is $\mathrm{O}(n)$ equivariant, we can see that $p_\omega(g|g' \cdot (\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top)) = p_\omega(g'^{-1}g|g'^{-1}g' \cdot (\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top)) = p_\omega(g'^{-1}g|\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top) = p_\omega(h|\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top)$. Thus we can rewrite the above expectation with respect to h as follows:

$$\begin{aligned}
\phi_{\theta,\omega}(g' \cdot \mathbf{x}) &= \mathbb{E}_{p_\omega(h|\mathbf{x}-\overline{\mathbf{x}}\mathbf{1}^\top)}\left[g' \cdot \bar{\mathbf{x}}\mathbf{1}^\top + g'h \cdot f_\theta((g'h)^{-1}g' \cdot (\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top))\right] + \mathbf{t}_{g'}\mathbf{1}^\top \\
&= \mathbb{E}_{p_\omega(h|\mathbf{x}-\overline{\mathbf{x}}\mathbf{1}^\top)}\left[g' \cdot \bar{\mathbf{x}}\mathbf{1}^\top + g'h \cdot f_\theta(h^{-1} \cdot (\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top))\right] + \mathbf{t}_{g'}\mathbf{1}^\top \\
&= \mathbf{Q}_{g'}\mathbb{E}_{p_\omega(h|\mathbf{x}-\overline{\mathbf{x}}\mathbf{1}^\top)}\left[\bar{\mathbf{x}}\mathbf{1}^\top + h \cdot f_\theta(h^{-1} \cdot (\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top))\right] + \mathbf{t}_{g'}\mathbf{1}^\top \\
&= \mathbf{Q}_{g'}\phi_{\theta,\omega}(\mathbf{x}) + \mathbf{t}_{g'}\mathbf{1}^\top \\
&= g' \cdot \phi_{\theta,\omega}(\mathbf{x}),
\end{aligned} \tag{34}$$

showing the $\mathrm{E}(n)$ equivariance of $\phi_{\theta,\omega}$. □

Proposition 7. *The proposed symmetrization $\phi_{\theta,\omega}$ for special Euclidean group $\mathrm{SE}(n)$ is equivariant.*

Proof. We prove $\phi_{\theta,\omega}(g' \cdot \mathbf{x}) = g' \cdot \phi_{\theta,\omega}(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{X}$ and $g' \in \mathrm{SE}(n)$, in an analogous manner to the proof of Proposition 6. From Eq. (29), we have:

$$\phi_{\theta,\omega}(g' \cdot \mathbf{x}) = \mathbb{E}_{p_\omega(g|g' \cdot \mathbf{x} - \overline{g' \cdot \mathbf{x}}\mathbf{1}^\top)}\left[\,\overline{g' \cdot \mathbf{x}}\mathbf{1}^\top + g \cdot f_\theta(g^{-1} \cdot (g' \cdot \mathbf{x} - \overline{g' \cdot \mathbf{x}}\mathbf{1}^\top))\right]. \tag{35}$$

In general, an element of special Euclidean group $g' \in \mathrm{SE}(n)$ acts on data $\mathbf{x} \in \mathbb{R}^{n \times d}$ via $g' \cdot \mathbf{x} = \mathbf{Q}^+_{g'}\mathbf{x} + \mathbf{t}_{g'}\mathbf{1}^\top$ where $\mathbf{Q}^+_{g'} \in \mathrm{SO}(n)$ is rotation component and $\mathbf{t}_{g'} \in \mathbb{R}^n$ is translation [69, 39]. With this, the centroid of the transformed data $g' \cdot \mathbf{x}$ is given as follows:

$$\overline{g' \cdot \mathbf{x}} = \overline{\mathbf{Q}^+_{g'}\mathbf{x} + \mathbf{t}_{g'}\mathbf{1}^\top} = \overline{\mathbf{Q}^+_{g'}\mathbf{x}} + \mathbf{t}_{g'} = \mathbf{Q}^+_{g'}\overline{\mathbf{x}} + \mathbf{t}_{g'}, \tag{36}$$

which leads to the following:

$$\begin{aligned}
g' \cdot \mathbf{x} - \overline{g' \cdot \mathbf{x}}\mathbf{1}^\top &= \mathbf{Q}^+_{g'}\mathbf{x} + \mathbf{t}_{g'}\mathbf{1}^\top - \mathbf{Q}^+_{g'}\overline{\mathbf{x}}\mathbf{1}^\top - \mathbf{t}_{g'}\mathbf{1}^\top \\
&= \mathbf{Q}^+_{g'}(\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top).
\end{aligned} \tag{37}$$

Similar as in Proposition 6, subtracting centroid only leaves $\mathrm{SO}(n)$ component. We then have:

$$\begin{aligned}
\phi_{\theta,\omega}(g' \cdot \mathbf{x}) &= \mathbb{E}_{p_\omega(g|\mathbf{Q}^+_{g'}(\mathbf{x}-\overline{\mathbf{x}}\mathbf{1}^\top))}\left[\mathbf{Q}^+_{g'}\bar{\mathbf{x}}\mathbf{1}^\top + \mathbf{t}_{g'}\mathbf{1}^\top + g \cdot f_\theta(g^{-1} \cdot (\mathbf{Q}^+_{g'}(\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top)))\right] \\
&= \mathbb{E}_{p_\omega(g|g' \cdot (\mathbf{x}-\overline{\mathbf{x}}\mathbf{1}^\top))}\left[g' \cdot \bar{\mathbf{x}}\mathbf{1}^\top + g \cdot f_\theta(g^{-1}g' \cdot (\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top))\right] + \mathbf{t}_{g'}\mathbf{1}^\top,
\end{aligned} \tag{38}$$

where, inside the expectation, we interpret the rotation component of g' as an element of the special orthogonal group $\mathrm{SO}(n)$. Similar as in Theorem 1, we introduce $h = g'^{-1}g \in \mathrm{SO}(n)$ that $g = g'h$. As the distribution p_ω is $\mathrm{SO}(n)$ equivariant, we have $p_\omega(g|g' \cdot (\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top)) = p_\omega(g'^{-1}g|g'^{-1}g' \cdot (\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top)) = p_\omega(h|\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top)$. We then rewrite the expectation with respect to h:

$$\begin{aligned}
\phi_{\theta,\omega}(g' \cdot \mathbf{x}) &= \mathbb{E}_{p_\omega(h|\mathbf{x}-\overline{\mathbf{x}}\mathbf{1}^\top)}\left[g' \cdot \bar{\mathbf{x}}\mathbf{1}^\top + g'h \cdot f_\theta((g'h)^{-1}g' \cdot (\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top))\right] + \mathbf{t}_{g'}\mathbf{1}^\top \\
&= \mathbb{E}_{p_\omega(h|\mathbf{x}-\overline{\mathbf{x}}\mathbf{1}^\top)}\left[g' \cdot \bar{\mathbf{x}}\mathbf{1}^\top + g'h \cdot f_\theta(h^{-1} \cdot (\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top))\right] + \mathbf{t}_{g'}\mathbf{1}^\top \\
&= \mathbf{Q}^+_{g'}\mathbb{E}_{p_\omega(h|\mathbf{x}-\overline{\mathbf{x}}\mathbf{1}^\top)}\left[\bar{\mathbf{x}}\mathbf{1}^\top + h \cdot f_\theta(h^{-1} \cdot (\mathbf{x} - \overline{\mathbf{x}}\mathbf{1}^\top))\right] + \mathbf{t}_{g'}\mathbf{1}^\top \\
&= \mathbf{Q}^+_{g'}\phi_{\theta,\omega}(\mathbf{x}) + \mathbf{t}_{g'}\mathbf{1}^\top \\
&= g' \cdot \phi_{\theta,\omega}(\mathbf{x}),
\end{aligned} \tag{39}$$

showing the $\mathrm{SE}(n)$ equivariance of $\phi_{\theta,\omega}$. □

Product Group $H \times K$ For the product group $H \times K$, we assume that the base representation for each element $g = (h, k)$ is given as a pair of representations $\rho(g) = (\rho(h), \rho(k))$. Without loss of generality, we further assume that the representation $\rho(g)$ can be expressed as the Kronecker product $\rho(g) = \rho(h) \otimes \rho(k)$ that acts on flattened data $\mathrm{vec}(\mathbf{x})$ as $\mathbf{x} \mapsto \mathrm{vec}^{-1}(\rho(g)\mathrm{vec}(\mathbf{x}))$. This follows the standard approach in equivariant deep learning [30, 55] that deals with composite representations using direct sum and tensor products of base group representations.

Above approach applies to many practical product groups, including sets and graphs with Euclidean attributes ($\mathrm{S}_n \times \mathrm{O}(d)/\mathrm{SO}(d)$[6]) and sets of symmetric elements ($\mathrm{S}_n \times H$) in general [57]. For

[6]This is after handling the translation component of the Euclidean group $\mathrm{E}(d)/\mathrm{SE}(d)$ as in Eq. (29).

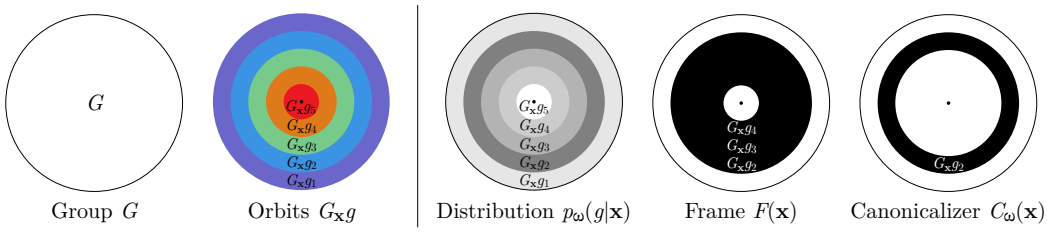

Figure 3: Visual illustration of the symmetrization methods based on probabilities assigned upon the partitioning of the group G into orbits $G_{\mathbf{x}}g$. Note that, while we use concentric circles of different perimeters to illustrate each orbit, all orbits actually have an identical cardinality $|G_{\mathbf{x}}g| = |G_{\mathbf{x}}|$.

example, for the group $\mathrm{S}_n \times \mathrm{O}(d)$ on data $\mathbf{x} \in \mathbb{R}^{n\times d}$, an element $g = (h, k)$ has representation $\rho(g) = \rho(h) \otimes \rho(k) \in \mathbb{R}^{nd\times nd}$ combined from permutation $\rho(h) \in \mathbb{R}^{n\times n}$ and rotation $\rho(k) \in \mathbb{R}^{d\times d}$, which acts by $\mathbf{x} \mapsto \mathrm{vec}^{-1}(\rho(g)\mathrm{vec}(\mathbf{x}))$ or more simply $\mathbf{x} \mapsto \rho(h)\mathbf{x}\rho(k)^\top$.

Now we recall that the $p_\omega(g|\mathbf{x})$ for the product group $H \times K$ is implemented as follows:

1. Sample noise $\boldsymbol{\epsilon} \in \mathcal{E}$ from i.i.d. normal $\mathcal{N}(0, \eta^2)$ such that $p(\boldsymbol{\epsilon})$ is invariant under faithful orthogonal representations of H and K. For example, for $\mathrm{S}_n \times \mathrm{O}(d)$, the noise $\boldsymbol{\epsilon} \in \mathbb{R}^{n\times d}$ that follows i.i.d. normal $\mathcal{N}(0, \eta^2)$ is invariant under base representations of both S_n and $\mathrm{O}(d)$.
2. Use a $H \times K$ equivariant neural network to obtain features $(\mathbf{x}, \boldsymbol{\epsilon}) \mapsto (\mathbf{Z}_H, \mathbf{Z}_K)$ where $\mathbf{Z}_H$ is K invariant and $\mathbf{Z}_K$ is H invariant. For example, for $\mathrm{S}_n \times \mathrm{O}(d)$, we expect node-level scalar features $\mathbf{Z}_{\mathrm{S}_n} \in \mathbb{R}^n$ to be $\mathrm{O}(d)$ invariant and d global rotary features $\mathbf{Z}_{\mathrm{O}(d)} \in \mathbb{R}^{d\times d}$ to be S_n invariant.
3. Apply postprocessing for H and K groups onto $\mathbf{Z}_H$ and $\mathbf{Z}_K$ respectively to obtain representations $\mathbf{Z}_H \mapsto \rho(h)$ and $\mathbf{Z}_K \mapsto \rho(k)$ of H and K groups respectively. For example, for $\mathrm{S}_n \times \mathrm{O}(d)$, we use argsort in Eq. (23) to obtain $\mathbf{Z}_{\mathrm{S}_n} \mapsto \rho(h)$ and Gram-Schmidt process to obtain $\mathbf{Z}_{\mathrm{O}(d)} \mapsto \rho(k)$.
4. Combine the representations $\rho(g) = (\rho(h), \rho(k))$ to obtain a representation for the $H \times K$ group.

We now show the following:

Proposition 8. *The proposed distribution $p_\omega(g|\mathbf{x})$ for the product group $H \times K$ is equivariant.*

Proof. By assumption, $p(\boldsymbol{\epsilon})$ is invariant under faithful orthogonal representations of H and K. This implies $H \times K$ invariance as well, since $p(\boldsymbol{\epsilon}) = p(h \cdot \boldsymbol{\epsilon}) = p(k \cdot \boldsymbol{\epsilon})$ for all $\boldsymbol{\epsilon} \in \mathcal{E}, h \in H, k \in K$ gives $p(k \cdot h \cdot \boldsymbol{\epsilon}) = p(k \cdot (h \cdot \boldsymbol{\epsilon})) = p(h \cdot \boldsymbol{\epsilon}) = p(\boldsymbol{\epsilon})$, and Kronecker product of faithful orthogonal representations gives a faithful orthogonal representation. Furthermore, the map $(\mathbf{x}, \boldsymbol{\epsilon}) \mapsto (\rho(h), \rho(k)) = \rho(g)$ is overall $H \times K$ equivariant, since an input transformed with $g' = (h', k')$ is first mapped by the equivariant neural network as $(g' \cdot \mathbf{x}, g' \cdot \boldsymbol{\epsilon}) \mapsto (h' \cdot \mathbf{Z}_H, k' \cdot \mathbf{Z}_K)$, then postprocessed as $(h' \cdot \mathbf{Z}_H, k' \cdot \mathbf{Z}_K) \mapsto (\rho(h')\rho(h), \rho(k')\rho(k)) = (\rho(h'), \rho(k')) \cdot (\rho(h), \rho(k)) = \rho(g')\rho(g)$. Combining the above, by Theorem 3, the distribution $p_\omega(g|\mathbf{x})$ is $H \times K$ equivariant. □

A.1.5 Proof of Proposition 1 and Proposition 2 (Section 2.4)

Before proceeding to proofs, we recall that the stabilizer subgroup $G_{\mathbf{x}}$ of a group G for $\mathbf{x}$ is defined as $\{g' \in G : g' \cdot \mathbf{x} = \mathbf{x}\}$ and acts on a given group element $g \in G$ through left multiplication $g \mapsto g'g$. For some $g \in G$, by $G_{\mathbf{x}}g$ we denote its orbit under the action by $G_{\mathbf{x}}$, *i.e.*, the set of elements in G to which g can be moved by the action of elements $g' \in G_{\mathbf{x}}$. Importantly, we can show the following:

Property 1. *Any group G is a union of disjoint orbits $G_{\mathbf{x}}g$ of equal cardinality.*

Proof. Let us consider the equivalence relation $\sim$ on G induced by the action of the stabilizer $G_{\mathbf{x}}$, defined as $g \sim h \iff h \in G_{\mathbf{x}}g$. The orbits $G_{\mathbf{x}}g$ are the equivalence classes under this relation, and the set of all orbits of G under the action of $G_{\mathbf{x}}$ forms a partition of G (*i.e.*, the quotient $G/G_{\mathbf{x}}$). Furthermore, since $G_{\mathbf{x}} \leq G$ and right multiplication by some $g \in G$ is a faithful action of G on itself, we have $|G_{\mathbf{x}}g| = |G_{\mathbf{x}}|$ for all $g \in G$, which shows that all orbits $G_{\mathbf{x}}g$ have equal cardinality. □

The partition of group G into disjoint orbits $G_{\mathbf{x}}g$ is illustrated in the first and second panel of Figure 3. We now show the following:

Property 2. *G equivariant $p_\omega(g|\mathbf{x})$ assigns identical probability to all elements on each orbit $G_\mathbf{x} g$.*

Proof. With equivariance, we have $p_\omega(g|\mathbf{x}) = p_\omega(g'g|g'\cdot\mathbf{x})$. Since $g'\cdot\mathbf{x} = \mathbf{x}$ for all $g' \in G_\mathbf{x}$, we have $p_\omega(g|\mathbf{x}) = p_\omega(g'g|\mathbf{x})$ for all $g' \in G_\mathbf{x}$; all elements on orbit $G_\mathbf{x} g$ have an identical probability. □

Property 2 characterizes probability distributions over G that can be expressed with $p_\omega(g|\mathbf{x})$, which we illustrate in the third panel of Figure 3. Intuitively, $p_\omega(g|\mathbf{x})$ assigns constant probability densities over each of the orbit $G_\mathbf{x} g$ that partitions G as shown in Property 1. We now prove Proposition 1 and Proposition 2 by showing that $p_\omega(g|\mathbf{x})$ can become frame and canonicalizer as special cases:

Proposition 1. *Probabilistic symmetrization with G equivariant distribution $p_\omega(g|\mathbf{x})$ can become frame averaging [69] by assigning uniform density to a set of orbits $G_\mathbf{x} g$ for some group elements g.*

Proof. A frame is defined as a set-valued function $F : \mathcal{X} \to 2^G \setminus \emptyset$ that satisfies G equivariance $F(g\cdot\mathbf{x}) = gF(\mathbf{x})$ [69]. For some frame F, frame averaging is defined as follows:

$$\frac{1}{|F(\mathbf{x})|} \sum_{g \in F(\mathbf{x})} \left[g \cdot f_\theta(g^{-1}\cdot\mathbf{x}) \right], \tag{40}$$

which can be equivalently written as the below expectation:

$$\mathbb{E}_{g\sim \text{Unif}(F(\mathbf{x}))} \left[g \cdot f_\theta(g^{-1}\cdot\mathbf{x}) \right]. \tag{41}$$

From Theorem 3 of [69], we have that $F(\mathbf{x})$ is a disjoint union of equal size orbits $G_\mathbf{x} g$. Therefore, $\text{Unif}(F(\mathbf{x}))$ is a uniform probability distribution over the union of the orbits. This can be expressed by a G equivariant distribution $p_\omega(g|\mathbf{x})$ by assigning identical probability over all orbits in the frame F and zero probability to all orbits not in the frame (illustrated in the fourth panel of Figure 3). Therefore, probabilistic symmetrization can become frame averaging. □

Proposition 2. *Probabilistic symmetrization with G equivariant distribution $p_\omega(g|\mathbf{x})$ can become canonicalization [39] by assigning uniform density to a single orbit $G_\mathbf{x} g$ of some group element g.*

Proof. A canonicalizer is defined as a (possibly stochastic) parameterized map $C_\omega : \mathcal{X} \to G$ that satisfies relaxed G equivariance $C_\omega(g\cdot\mathbf{x}) = gg'C_\omega(\mathbf{x})$ for some $g' \in G_\mathbf{x}$ [39]. For some canonicalizer C_ω, canonicalization is defined as follows:

$$g \cdot f_\theta(g^{-1}\cdot\mathbf{x}), \quad g = C_\omega(\mathbf{x}). \tag{42}$$

From relaxed G equivariance, we have $C_\omega(\mathbf{x}) = g'C_\omega(\mathbf{x})$ for some $g' \in G_\mathbf{x}$. A valid choice for the canonicalizer C_ω is a stochastic map that samples from the uniform distribution over a frame $C_\omega(\mathbf{x}) \sim \text{Unif}(F_\omega(\mathbf{x}))$ where the frame is assumed to always provide a single orbit $F_\omega(\mathbf{x}) = G_\mathbf{x} g$. In this case, canonicalization is equivalent to a 1-sample estimation of the below expectation:

$$\mathbb{E}_{g\sim \text{Unif}(F_\omega(\mathbf{x}))} \left[g \cdot f_\theta(g^{-1}\cdot\mathbf{x}) \right]. \tag{43}$$

Furthermore, uniform distribution over the single-orbit frame $\text{Unif}(F_\omega(\mathbf{x}))$ can be expressed by a G equivariant distribution $p_\omega(g|\mathbf{x})$ by assigning nonzero probability to the single orbit $G_\mathbf{x} g$ and assigning zero probability to the rest (illustrated in the last panel of Figure 3). Therefore, probabilistic symmetrization can become canonicalization. □

A.2 Extended Related Work (Continued from Section 2.4)

Our work draws inspiration from an extensive array of prior research, ranging from equivariant architectures and symmetrization to general-purpose deep learning with transformers. This section outlines a comprehensive review of these fields, spotlighting ideas specifically relevant to our work.

Equivariant Architectures Equivariant architectures, defined by the group equivariance of their building blocks, have been a prominent approach for equivariant deep learning [12, 10]. These architectures have been primarily developed for data types associated with permutation and Euclidean group symmetries, including images [18, 19], sets, graphs, and hypergraphs [6, 55, 8], and geometric graphs [23, 76, 81]. Additionally, they have been extended to more general data types under arbitrary finite group [72] and matrix group symmetries [30]. However, they face challenges such as limited expressive power [89, 54, 60, 93, 38] and architectural issues like over-smoothing [66, 14, 65] and over-squashing [82] in graph neural networks. Our work aims to develop an equivariant deep learning approach that relies less on equivariant architectures, to circumvent these limitations and enhance parameter sharing and transfer across varying group symmetries.

Table 5: Overview of the datasets.

Dataset	Symmetry	Domain	Task	Feat. (dim)
GRAPH8c EXP EXP-classify	S_n Invariant	Graph Isomorphism	Graph Separation Graph Classification	Adj. (1)
n-body	$S_n \times E(3)$ Equivariant	Physics	Position Regression	Pos. (3) + Vel. (3) + Charge (1)
PATTERN	S_n Equivariant	Mathematical Modeling	Node Classification	Rand. Node Attr. (3) + Adj. (1)
Peptides-func Peptides-struct	S_n Invariant	Chemistry	Graph Classification Graph Regression	Atom (9) + Bond (3) + Adj. (1)
PCQM-Contact	S_n Equivariant	Quantum Chemistry	Link Prediction	Atom (9) + Bond (3) + Adj. (1)

Symmetrization Our approach is an instance of symmetrization for equivariant deep learning which aims to achieve group equivariance using base models with unconstrained architectures. This is in general accomplished by averaging over specific group transformations of the input and output such that the averaged output exhibits equivariance. This allows us to leverage the expressive power of the base model *e.g.*, achieve universal approximation using an MLP [34, 20] or a transformer [92], and potentially share or transfer parameters across different group symmetries. Existing literature has explored the choices of group transformations and base models for symmetrization. A straightforward approach is to average over the entire group [90], which is suitable for small, finite groups [5, 61, 40, 83] and requires sampling-based estimation for large groups such as permutations [63, 64, 80, 21]. Recent studies have attempted to identify smaller, input-dependent subsets of the group for averaging. Frame averaging [69] employs manually discovered input-dependent subsets, which still demand sampling-based estimation for certain worst-case inputs. Canonicalization [39] utilizes a single group transformation predicted by a neural network, but sacrifices strict equivariance. Our approach jointly achieves equivariance and end-to-end learning by utilizing parameterized, input-conditional equivariant distributions. Furthermore, our approach is one of the first demonstrations of symmetrization for the permutation group in real-world graph recognition task. Concerning the base model, previous work mostly examined small base models like an MLP or partial symmetrization of already equivariant models like GNNs. Few studies have explored symmetrizing pre-trained models for small finite groups [5, 4], and to our knowledge, we are the first to investigate symmetrization of a pre-trained standard transformer for permutation groups or any large group generally.

Transformer Architectures A significant motivation of our work is to combine the powerful scaling and transfer capabilities of the standard transformer architecture [84] with equivariant deep learning. The transformer architecture has driven major breakthroughs in language and vision domains [84, 24, 13, 70], and proven its ability to learn diverse modalities [37, 36] or transfer knowledge across them [77, 52, 74, 25, 67, 50]. Although transformer-style architectures have been developed for symmetric data modalities like sets [49], graphs [91, 43, 41, 48, 62, 59], hypergraphs [17, 42], and geometric graphs [31, 53], they often require specific architectural modifications to achieve equivariance to the given symmetry group, compromising full compatibility with transformer architectures used in language and vision domains. Apart from a few studies on linguistic graph encoding with language models [73], we believe we are the first to propose a general framework that facilitates full compatibility of the standard transformer architecture for learning symmetric data. For example, we have shown that a pre-trained vision transformer could be repurposed to encode graphs.

A.3 Experimental Details (Section 3)

We provide details of the datasets and models used in our experiments in Section 3. The details of the datasets from the original papers [2, 1, 27, 28, 31, 76] can be found in Table 5 and Table 6.

A.3.1 Implementation Details of p_ω for Symmetric Group S_n (Section 3.1, 3.3, 3.4)

In all experiments regarding the symmetric group S_n, we implemented the S_n equivariant distribution $p_\omega(g|\mathbf{x})$, *i.e.*, $q_\omega : (\mathbf{x}, \boldsymbol{\epsilon}) \mapsto \mathbf{P}_g$ based on a 3-layer GIN [89]. Specifically, given a graph $\mathbf{x}$ with node features $\mathbf{X} \in \mathbb{R}^{n \times d_{\text{in}}}$ and adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$,[7] we first augment a virtual node [32] which

[7]We do not utilize edge attributes in equivariant distribution p_ω, while we utilize them in base model f_θ.

Table 6: Statistics of the datasets.

Dataset	Size	Max # Nodes	Average # Nodes	Average # Edges
GRAPH8c	11,117	8	8	28.82
EXP EXP-classify	1,200	64	44.44	110.21
n-body	7,000	5	5	Fully Connected
PATTERN	14,000	188	117.47	4749.15
Peptides-func Peptides-struct	15,535	444	150.94	307.30
PCQM-Contact	529,434	53	30.14	61.09

is connected to all nodes to facilitate global interaction while retaining S_n equivariance, as follows:

$$\mathbf{X}' = [\mathbf{X}; \mathbf{v}], \quad \mathbf{A}' = \begin{bmatrix} \mathbf{A} & \mathbf{1} \\ \mathbf{1}^\top & 0 \end{bmatrix}, \tag{44}$$

where the feature of the virtual node $\mathbf{v} \in \mathbb{R}^{d_{\text{in}}}$ is a trainable parameter. Then, we prepared the input node features $\mathbf{H} \in \mathbb{R}^{(n+1)\times d_{\text{in}}}$ to the GIN as $\mathbf{H} = \mathbf{X}' + \boldsymbol{\epsilon}$ where the noise $\boldsymbol{\epsilon} \in \mathbb{R}^{(n+1)\times d_{\text{in}}}$ is i.i.d. sampled from Unif$[0, \eta]$ with scale hyperparameter η. Then, we employ following 3-layer GIN with hidden dimension d to obtain processed node features $\mathbf{H}' \in \mathbb{R}^{(n+1)\times 1}$:

$$\mathbf{H}' = \text{GINConv}_{d,d,1} \circ \text{GINConv}_{d,d,d} \circ \text{GINConv}_{d_{\text{in}},d,d}(\mathbf{H}), \tag{45}$$

where each $\text{GINConv}_{d_1,d_2,d_3}$ computes the following with a two-layer elementwise MLP : $\mathbb{R}^{n\times d_1} \to \mathbb{R}^{n\times d_3}$ with hidden dimension d_2, ReLU activation, batch normalization, and trained scalar e:

$$\mathbf{H} \mapsto \text{MLP}((\mathbf{A}' + (1+e)\mathbf{I})\mathbf{H}). \tag{46}$$

Then, from the processed node features $\mathbf{H}' \in \mathbb{R}^{(n+1)\times 1}$, we finally obtain the features $\mathbf{Z} \in \mathbb{R}^n$ for postprocessing by discarding the feature of the virtual node. Then, postprocessing with argsort : $\mathbf{Z} \mapsto \mathbf{P}_g \in \mathbb{R}^{n\times n}$ is performed identically as in the main text (Section 2.2) with temperature $\tau = 0.01$. In addition, we use a regularizer on the sum of entropy of each row and column of the soft permutation matrix $\hat{\mathbf{P}}_g$ (Eq. (8)) with strength 0.1 in all experiments, which encourages learning sharper $\hat{\mathbf{P}}_g$ [86].

A.3.2 Implementation Details of p_ω for Product Group $\mathrm{S}_n \times \mathrm{E}(3)$ (Section 3.2)

In our n-body experiment on the product group $\mathrm{S}_n \times \mathrm{E}(3)$, we implemented the $\mathrm{S}_n \times \mathrm{O}(3)$ equivariant distribution $p_\omega(g|\mathbf{x} - \bar{\mathbf{x}}\mathbf{1}^\top)$, *i.e.*, $q_\omega : (\mathbf{x} - \bar{\mathbf{x}}\mathbf{1}^\top, \boldsymbol{\epsilon}) \mapsto (\mathbf{P}_g, \mathbf{Q}_g)$ based on a 2-layer Vector Neurons version of DGCNN with 96 hidden dimensions [23]. Due to its complexity, we focus on describing input and output of the network and postprocessing, and guide the readers to the original paper [23] for further architectural details. In a high-level, the Vector Neurons receives position $\mathbf{P} \in \mathbb{R}^{n\times 3}$ and velocity $\mathbf{V} \in \mathbb{R}^{n\times 3}$ of the zero-centered input $\mathbf{x} - \bar{\mathbf{x}}\mathbf{1}^\top$ with noises $\boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_2 \in \mathbb{R}^{n\times 3}$ i.i.d. sampled from normal $\mathcal{N}(0, \eta^2)$ with scale hyperparameter η, and produces features $\mathbf{H}_{\mathrm{S}_n} \in \mathbb{R}^{n\times 3\times d_1}$ and $\mathbf{H}_{\mathrm{O}(3)} \in \mathbb{R}^{n\times 3\times d_2}$ with $d_1 = 1$ and $d_2 = 3$ as follows:

$$\mathbf{H}_{\mathrm{S}_n}, \mathbf{H}_{\mathrm{O}(3)} = \text{VN-DGCNN}(\mathbf{P} + \boldsymbol{\epsilon}_1, \mathbf{V} + \boldsymbol{\epsilon}_2). \tag{47}$$

Then, we apply $\mathrm{O}(3)$ invariant pooling on $\mathbf{H}_{\mathrm{S}_n}$ and S_n invariant pooling on $\mathbf{H}_{\mathrm{O}(3)}$, both supported as a part of [23], to obtain features for postprocessing $\mathbf{Z}_{\mathrm{S}_n} \in \mathbb{R}^{n\times 1}$ and $\mathbf{Z}_{\mathrm{O}(3)} \in \mathbb{R}^{3\times 3}$, respectively:

$$\mathbf{Z}_{\mathrm{S}_n} = \text{Pool}_{\mathrm{O}(3)}(\mathbf{H}_{\mathrm{S}_n}), \quad \mathbf{Z}_{\mathrm{O}(3)} = \text{Pool}_{\mathrm{S}_n}(\mathbf{H}_{\mathrm{O}(3)}). \tag{48}$$

Then, postprocessing with argsort : $\mathbf{Z}_{\mathrm{S}_n} \mapsto \mathbf{P}_g \in \mathbb{R}^{n\times n}$ and Gram-Schmidt orthogonalization $\mathbf{Z}_{\mathrm{O}(3)} \mapsto \mathbf{Q}_g \in \mathbb{R}^{3\times 3}$ is performed identically as described in the main text (Section 2.2). For the argsort operator, similar as in Appendix A.3.1, we use temperature $\tau = 0.01$ and entropy regularizer on the soft permutation matrix $\hat{\mathbf{P}}_g$ with strength 0.1.

A.3.3 Graph Isomorphism Learning with MLP (Section 3.1)

Base Model f_θ For EXP and EXP-classify, the model is given adjacency matrix $\mathbf{A} \in \mathbb{R}^{64\times 64}$ and binary node features $\mathbf{X} \in \mathbb{R}^{64}$ which are zero-padded to maximal number of nodes 64. For GRAPH8c, the input graphs are all of size 8 without node features, and the model is given adjacency matrix $\mathbf{A} \in \mathbb{R}^{8\times 8}$. For EXP-classify, the prediction target is a scalar binary classification logit.

For the base model for EXP-classify, we use a 5-layer MLP $f_\theta : \mathbb{R}^{64\times 64+64} \to \mathbb{R}$ on flattened and concatenated adjacency matrix and node features, with an identical architecture to other symmetrization baselines (MLP-GA and MLP-FA [69]) as in below:

$$f_\theta = \mathrm{FC}_{1,10} \circ \mathrm{FC}_{10,2048} \circ \mathrm{FC}_{2048,4096} \circ \mathrm{FC}_{4096,2048} \circ \mathrm{FC}_{2048,4160}, \tag{49}$$

where $\mathrm{FC}_{d_2,d_1} : \mathbb{R}^{d_1} \to \mathbb{R}^{d_2}$ denotes a fully-connected layer and ReLU activation is omitted. For EXP, we drop the last layer to obtain 10-dimensional output. For GRAPH8c, we use the following architecture $f_\theta : \mathbb{R}^{8\times 8} \to \mathbb{R}^{10}$ that takes flattened adjacency to produce 10-dimensional output [69]:

$$f_\theta = \mathrm{FC}_{10,64} \circ \mathrm{FC}_{64,128} \circ \mathrm{FC}_{128,64}. \tag{50}$$

Training For EXP-classify, we train our models with binary cross-entropy loss using Adam optimizer [44] with batch size 100 and learning rate 1e-3 for 1,000 epochs, which takes around 10 minutes on a single RTX 3090 GPU with 24GB using PyTorch [68]. We additionally apply gradient norm clipping at 0.1, which we found helpful for stabilizing the training. For the equivariant distribution p_ω, we use noise scale $\eta = 1$. Since EXP and GRAPH8c concern randomly initialized models, we do not train the models for these tasks.

A.3.4 Particle Dynamics Learning with Transformer (Section 3.2)

Base Model f_θ The model is given zero-centered position $\mathbf{P} \in \mathbb{R}^{5\times 3}$ and velocity $\mathbf{V} \in \mathbb{R}^{5\times 3}$ of 5 particles at a time point with pairwise charge difference $\mathbf{C} \in \mathbb{R}^{5\times 5}$ and squared distance $\mathbf{D} \in \mathbb{R}^{5\times 5}$. We set the prediction target as difference of position $\Delta\mathbf{P} \in \mathbb{R}^{5\times 3}$ after a certain time.

For the base model, we use a 4-layer transformer encoder $f_\theta : \mathbb{R}^{25\times 8} \to \mathbb{R}^{25\times 3}$ that operates on sequences of length 25 with dimension 8. At each prediction, we first organize the input into a single tensor $\in \mathbb{R}^{5\times 5\times 8}$ by placing $\mathbf{P}$ and $\mathbf{V}$ on the diagonals of $\mathbf{C}$ and $\mathbf{D}$, and then turn the tensor into a sequence of 25 tokens $\in \mathbb{R}^{25\times 8}$ by flattening the first two axes. For the transformer, we use the standard implementation provided in PyTorch [68, 84], with 64 hidden dimensions, 4 attention heads, GELU activation [33] in feedforward networks, PreLN [88], and learnable 1D positional encoding. The model has around 105k trainable parameters, slightly larger than the backbones of symmetrization baselines in the benchmark (GNN-FA and GNN-Canonical.) with 80k parameters.

Training We train our models with MSE loss using Adam optimizer [44] with batch size 64 and learning rate 1e-4 for 10,000 epochs, which takes around 8.5 hours on a single RTX 3090 GPU with 24GB using PyTorch [68]. We additionally use weight decay with strength 1e-8, gradient norm clipping at 0.1, and dropout regularization only on the distribution p_ω with probability 0.08. For the equivariant distribution p_ω, we use noise scale $\eta = 1$.

A.3.5 Graph Pattern Recognition with Vision Transformer (Section 3.3)

Base Model f_θ The model is given adjacency matrix $\mathbf{A} \in \mathbb{R}^{188\times 188}$ and node features $\mathbf{X} \in \mathbb{R}^{188\times 3}$ zero-padded to maximal nodes 188. The prediction target is node classification logits $\mathbf{Y} \in \mathbb{R}^{188\times 2}$.

For the base model, we use a transformer with an identical architecture to ViT-Base [26] that operates on 224×224 images with 16×16 patch, using configuration from HuggingFace [87] model hub. We first remove the input patch projection and output head layers, which gives us a backbone transformer : $\mathbb{R}^{196\times 768} \to \mathbb{R}^{196\times 768}$ on sequences of length $(224/16) \times (224/16) = 14 \times 14 = 196$ tokens. Then, we use the following as the base model $f_\theta : (\mathbf{A}, \mathbf{X}) \mapsto \mathbf{Y}$:

$$f_\theta(\mathbf{A}, \mathbf{X}) = \mathrm{detokenize}_{1\mathrm{D}}\left(\mathrm{transformer}\left(\mathrm{tokenize}_{2\mathrm{D}}(\mathbf{A}) + \mathrm{tokenize}_{1\mathrm{D}}(\mathbf{X})\right)\right), \tag{51}$$

where $\mathrm{tokenize}_{2\mathrm{D}} : \mathbb{R}^{188\times 188\times 1} \to \mathbb{R}^{(14\times 14)\times 768}$ is 2D convolution with kernel and stride 14 and padding 8, followed by the flattening of spatial dimensions, $\mathrm{tokenize}_{1\mathrm{D}} : \mathbb{R}^{188\times 3} \to \mathbb{R}^{196\times 768}$ is 1D convolution with kernel and stride 1 and padding 8, and $\mathrm{detokenize}_{1\mathrm{D}} : \mathbb{R}^{196\times 768} \to \mathbb{R}^{188\times 2}$ is transposed 1D convolution with kernel and stride 1 and padding 8.

Note that the above (de)tokenizers treat the input and output as plain 1D or 2D array data, and apply linear projections of flattened patches for (de)tokenization with $14 \times 14 = 196$ tokens. We omitted input and output dimensions as they are clear from context.

Training We train our models with binary cross-entropy loss weighted inversely by class size [27] using Adam [44] optimizer with batch size searched from $\{128, 512\}$ and learning rate 1e-5 for the transformer and 1e-4 for the tokenizer, detokenizer, and distribution p_ω. We train the models for 50k steps under learning rate warm-up for 5k steps then *poly* decay [51] with early stopping based on validation loss, which usually takes less than 3 hours on 8 RTX 3090 GPUs with 24GB using PyTorch Lightning [29]. For the equivariant distribution p_ω we use noise scale $\eta = 1$, and for any symmetrization involving sampled average, we set sample size during training to 1 for batch size 512 and 10 for batch size 128 considering GPU memory cost.

A.3.6 Real-World Graph Learning with Vision Transformer (Section 3.4)

Base Model f_θ For Peptides-func/struct, the model is given adjacency matrix $\mathbf{A} \in \mathbb{R}^{444\times444}$, node features $\mathbf{X} \in \mathbb{R}^{444\times64}$, and edge features $\mathbf{E} \in \mathbb{R}^{444\times444\times7}$, zero-padded to maximal nodes 444. The prediction target is binary classification logits $\mathbf{Y} \in \mathbb{R}^{10}$ for Peptides-func, and regression targets $\mathbf{Y} \in \mathbb{R}^{11}$ for Peptides-struct. For PCQM-Contact, the model is given adjacency matrix $\mathbf{A} \in \mathbb{R}^{53\times53}$, node features $\mathbf{X} \in \mathbb{R}^{53\times68}$, and edge features $\mathbf{E} \in \mathbb{R}^{53\times53\times6}$, zero-padded to maximal nodes 53. The prediction target is binary edge classification logit $\mathbf{Y} \in \mathbb{R}^{53\times53\times1}$.

For the base model, we use a transformer with an identical architecture to ViT-Base [26], same as Appendix A.3.5. For Peptides-func/struct, we use the following as base model $f_\theta : (\mathbf{A}, \mathbf{X}, \mathbf{E}) \mapsto \mathbf{Y}$:

$$f_\theta(\mathbf{A}, \mathbf{X}, \mathbf{E}) = \text{detokenize}_{\texttt{[cls]}}\left(\text{transformer}\left(\text{tokenize}_{\text{2D}}(\mathbf{A}, \mathbf{E}) + \text{tokenize}_{\text{1D}}(\mathbf{X})\right)\right), \tag{52}$$

where $\text{tokenize}_{\text{2D}} : \mathbb{R}^{444\times444\times(1+7)} \to \mathbb{R}^{(14\times14)\times768}$ is 2D convolution with kernel and stride 32 and padding 4, followed by the flattening of spatial dimensions, $\text{tokenize}_{\text{1D}} : \mathbb{R}^{444\times64} \to \mathbb{R}^{196\times768}$ is 1D convolution with kernel and stride 3 and padding 144, and $\text{detokenize}_{\texttt{[cls]}}$ is linear projection of the `[cls]` token [26] to the target dimensionality. For PCQM-Contact, we use the following as base model $f_\theta : (\mathbf{A}, \mathbf{X}, \mathbf{E}) \mapsto \mathbf{Y}$:

$$f_\theta(\mathbf{A}, \mathbf{X}, \mathbf{E}) = \text{detokenize}_{\text{2D}}\left(\text{transformer}\left(\text{tokenize}_{\text{2D}}(\mathbf{A}, \mathbf{E}) + \text{tokenize}_{\text{1D}}(\mathbf{X})\right)\right), \tag{53}$$

where $\text{tokenize}_{\text{2D}} : \mathbb{R}^{53\times53\times(1+6)} \to \mathbb{R}^{(14\times14)\times768}$ is 2D convolution with kernel and stride 4 and padding 3, followed by the flattening of spatial dimensions, $\text{tokenize}_{\text{1D}} : \mathbb{R}^{53\times64} \to \mathbb{R}^{196\times768}$ is 1D convolution with kernel and stride 1 and padding 143, and $\text{detokenize}_{\text{2D}} : \mathbb{R}^{(14\times14)\times768} \to \mathbb{R}^{53\times53\times1}$ is transposed 2D convolution with kernel and stride 4 and padding 3, preceded by the unflattening of spatial dimensions.

Training We train our models with cross-entropy for classification and MAE for regression using Adam [44] optimizer with batch size 128 and learning rate 1e-5 for the transformer and 1e-4 for the tokenizer, detokenizer, and distribution p_ω. We train the models for 50k steps under learning rate warm-up for 5k steps then *poly* decay [51] with early stopping based on validation loss, which usually takes less than 3 hours on 8 RTX 3090 GPUs with 24GB using PyTorch Lightning [29]. For the equivariant distribution p_ω, we use noise scale $\eta = 1$.

A.4 Limitations and Broader Impacts

While the equivariance, universality, simplicity, and scalability of our approach offers a potential for positive impact for deep learning for chemistry, biology, physics, and mathematics, it also has limitations and potential negative impacts. The main limitation of our work is that it trades off certain desirable traits in equivariant deep learning in favor of achieving architecture agnostic equivariance. For example, **(1)** our approach is less interpretable compared to equivariant architectures due to less structured computations in the base model, **(2)** our approach is presumably less parameter efficient compared to equivariant architectures due to less imposed prior knowledge on parameterization, and **(3)** our approach is expected to be challenged when input size generalization is required, partially because the maximum input size has to be specified in advance. These limitations might lead to potential negative environmental impacts, since less interpretability and lower parameter efficiency implies higher reliance on larger models with more training cost. We acknowledge the aforementioned limitations and impacts of our work, and will make effort to address them in follow-up research.