Adaptive Canonicalization with Application to Invariant Anisotropic Geometric Networks

Anonymous authors
Paper under double-blind review

Abstract

Canonicalization is a widely used strategy in equivariant machine learning, enforcing symmetry in neural networks by mapping each input to a standard form. Yet, it often introduces discontinuities that can affect stability during training, limit generalization, and complicate universal approximation theorems. In this paper, we address this by introducing *adaptive canonicalization*, a general framework in which the canonicalization depends both on the input and the network. Specifically, we present the adaptive canonicalization based on prior maximization, where the standard form of the input is chosen to maximize the predictive confidence of the network. We prove that this construction yields continuous and symmetry-respecting models that admit universal approximation properties.

We propose two applications of our setting: (i) resolving eigenbasis ambiguities in spectral graph neural networks, and (ii) handling rotational symmetries in point clouds. We empirically validate our methods on molecular and protein classification, as well as point cloud classification tasks. Our adaptive canonicalization outperforms the three other common solutions to equivariant machine learning: data augmentation, standard canonicalization, and equivariant architectures.

1 Introduction

Equivariant machine learning (Gerken et al., 2023; Villar et al., 2021; Han et al., 2022; Keriven & Peyré, 2019) has been accentuated in geometric representation learning (Bronstein et al., 2017), motivated by the need to build models that respect symmetry inherent in data. For example, permutation equivariance in graphs (Gilmer et al., 2017; Zaheer et al., 2017; Xu et al., 2018), translation equivariance in images (LeCun & Bengio, 1998; Cohen & Welling, 2016a), and SO(3) or SE(3) equivariance for 3D objects and molecules (Thomas et al., 2018; Fuchs et al., 2020; Batzner et al., 2022; Satorras et al., 2021). The symmetry is built into the method so that transforming the input induces a predictable transformation of the output. This inductive bias reduces sample complexity, curbs overfitting to arbitrary poses, and often improves robustness on distribution shifts where the same object appears in a different orientation or ordering (Kondor & Trivedi, 2018; Wang et al., 2022; Park et al., 2022; Bronstein et al., 2021; Bietti & Mairal, 2019; Kaba & Ravanbakhsh, 2023).

There are three principal approaches to handling symmetry in machine learning. The first involves designing equivariant architectures (Cohen & Welling, 2016b; Weiler et al., 2018a; Weiler & Cesa, 2019; Geiger & Smidt, 2022; Maron et al., 2019a; Lippmann et al., 2024): neural network layers are constructed to commute with the symmetry. The second approach is data augmentation, where each datapoint is presented to the model at an arbitrary pose (Chen et al., 2020; Brandstetter et al., 2022). The third strategy is canonicalization (Kaba et al., 2023; Ma et al., 2023; 2024; Lim et al., 2022; 2023; Mondal et al., 2023; Lawrence et al., 2025; Sareen et al., 2025; Luo et al., 2022): each input is mapped to a standard form and then processed by a non-equivariant network. Another common approach to equivariant machine learning is frame averaging (Puny et al., 2021), which averages the network's output over a set of input transformations.

A well-known problem in canonicalization is that in many cases it unavoidably leads to an end-to-end architecture which is discontinuous with respect to the input (Dym et al., 2024; Zhang et al., 2019a; Lim et al., 2022). This inevitably leads to problems in stability during training and in generalization, as very similar inputs can lead to very different outputs (Dym et al., 2024; Tahmasebi & Jegelka, 2025a;b). Moreover, the discontinuity of the network makes universal approximation properties less

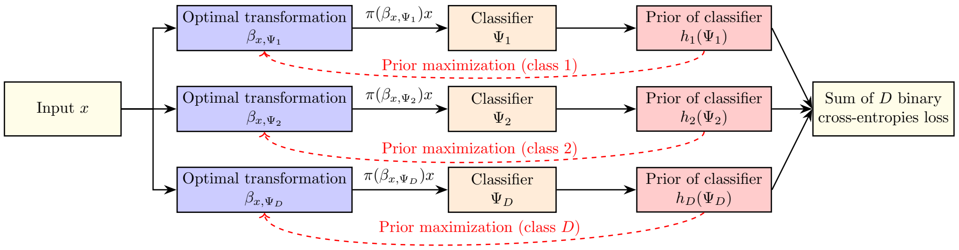
Input x
Optimal transformation β_{x,Ψ_1}
$\pi(\beta_{x,\Psi_1})x$
Classifier Ψ_1
Prior of classifier $h_1(\Psi_1)$
Prior maximization (class 1)
Optimal transformation β_{x,Ψ_2}
$\pi(\beta_{x,\Psi_2})x$
Classifier Ψ_2
Prior of classifier $h_2(\Psi_2)$
Prior maximization (class 2)
Optimal transformation β_{x,Ψ_D}
$\pi(\beta_{x,\Psi_D})x$
Classifier Ψ_D
Prior of classifier $h_D(\Psi_D)$
Prior maximization (class D)
Sum of D binary cross-entropies loss

Figure 1: Illustration of prior maximization adaptive canonicalization in classification. The adaptive canonicalization optimizes the transformations β_{x,Ψ_j} of the inputs x to the classifiers Ψ_j, while, during training, Ψ_j are simultaneously trained w.r.t. the adaptively canonicalized inputs $\pi(\beta_{x,\Psi_j})x$.

natural, as one approximates continuous symmetry preserving functions with discontinuous networks (Dym et al., 2024; Kaba et al., 2023; Wagstaff et al., 2022).

Our Contribution. In this paper, we show that the continuity problem in canonicalization can be solved if, instead of canonicalizing only as a function of the input, one defines a canonicalization that depends both on the input and the network. We propose such a general setting, which we call *adaptive canonicalization*, and show that it leads to continuous end-to-end models that respect the symmetries of the data and have universal approximation properties. Our theory does not only lead to superior theoretical properties w.r.t. standard canonicalization, but often also to superior empirical performance, specifically, in molecular, protein, and point cloud classification.

We focus on a specific class of adaptive canonicalizations that we call *prior maximizers*. To explain these methods, we offer the following illustrative example. Suppose that we would like to train a classifier of images into *cats*, *dogs* and *horses*. Suppose as well that each image x can appear in the dataset in any orientation, i.e., as $\pi(\alpha)x$ for any $\alpha \in [0, 2\pi]$ where $\pi(\alpha)$ is rotation by α. One standard approach for respecting this symmetry is to design an *equivariant architecture* Θ, which gives the same class probabilities to all rotations of the same image, i.e., $\Theta(\pi(\alpha)x) = \Theta(\pi(\alpha')x)$ for any to angles α, α'. Another simple approach for improving the classifier is to train a symmetryless network Ψ, and *augment* the training set with random rotations $\pi(\alpha)x$ for each input x. Yet another standard approach is to *canonicalize* the input, namely, to rotate each input image x by an angle β_x that depends on x in such a way that all rotated versions of the same image would have the exact same standard form, i.e., $\pi(\beta_{\pi(\alpha)x})\pi(\alpha)x = \pi(\beta_{\pi(\alpha')x})\pi(\alpha')x$ for any two angles α, α'. Then, the canonicalized image $\pi(\beta_x)x$ is plugged into a standard symmetryless neural network Ψ, and the end-to-end architecture $\Psi(\pi(\beta_x)x)$ is guaranteed to be invariant to rotations. We propose a fourth approach, where the canonicalized rotation $\pi(\beta_{x,\Psi})$ depends both on the image x and on the (symmetryless) neural network Ψ.

To motivate this approach, consider a neural network Ψ which, by virtue of being symmetriless, may perform better on some orientations of x than others. For illustration, it is easier for humans to detect an image as a horse if its limbs point downwards. Suppose that $\Psi(x) = (\Psi_j(x))_{j=1}^3 = (\Psi_{dog}(x), \Psi_{cat}(x), \Psi_{horse}(x))$ is a sequence of binary classifiers with values in $[0, 1]$ each. The output of Ψ is defined to be the class with highest probability. Suppose moreover that for each j, the network Ψ_j is granted the ability to rotated x freely, and probe the output $\Psi_j(\pi(\alpha)x)$ for each α. The network then chooses the orientation α_* such that $\Psi_j(\pi(\alpha_*)x)$ is maximized. As an analogy, one can imagine an image on a piece of paper being handed at a random orientation to a person with visual system Ψ. To detect if there is a horse in the image, the person would rotate the paper, searching for an orientation which looks like a horse. Namely, if there is an orientation α_* where $\Psi_{horse}(\pi(\alpha_*)x)$ is high then there is a horse in the image, and otherwise there is non. This process would be repeated for all other classes, and eventually the image would be classified as the j_* such that $\max_\alpha \Psi_{j_*}(\pi(\alpha)x)$ is greater than $\max_\alpha \Psi_j(\pi(\alpha)x)$ for all other $j \neq j_*$. This is the process that we call *prior maximization adaptive canonicalization*. This process is inspired by ideas from cognitive psychology, where the human visual system is believed to learn canonical mental models of objects and to discard redundant variation due to symmetries by mentally "rotating" perceived stimuli into alignment with these canonical views (Shepard & Metzler, 1971; Cooper & Shepard, 1973; Tarr & Pinker, 1989). Our example of a person rotating a sheet of paper to recognize whether it contains a horse is directly inspired by this line of work: prior maximization adaptive canonicalization can be viewed as a neural analogue of mental rotation, where the network searches over transformations to align inputs with its learned canonical views. We note that some previous models in machine learning were also inspired by this process (Palmer, 1981; Harris et al., 2001; Graf, 2006; Gomez et al., 2008; Konkle & Oliva, 2011; Risko & Gilbert, 2016; Tacchetti et al., 2018; Schmidt & Stober, 2024).

We note that the inputs are adaptively canonicalized also during training, so Ψ needs not learn to respect any symmetry on its own. In fact, Ψ can benefit from being symmetryless. For example, it may search for "horse head" patterns only diagonally above where it detects "horse limbs" patterns, and rely on the prior maximization to orient horses accordingly.

We show that adaptive canonicalization leads to a continuous symmetry preserving end-to-end classifier that can approximate any symmetry preserving continuous function when Ψ are non-equivariant neural networks. As an application, we propose adaptive canonicalization methods for 1) spectral graph neural networks, where the symmetry is in the choice of the eigenbasis of the graph shift operator, and 2) point clouds, with rotation symmetries. We show that adaptive canonicalization in these cases outperforms both standard canonicalization and equivariant networks, as well as augmentation methods. See Fig. 1 for an illustration of prior maximization.

2 Related Work

Canonicalization has been studied in several forms. For example, in computer vision and geometric deep learning methods, inputs are often first transformed into a standardized pose or reference frame before classification (Lowe, 2004; Jaderberg et al., 2015). More recent work formalizes this as an explicit canonicalization map feeding a downstream network (Lim et al., 2022; Ma et al., 2024) or as energy-based canonicalization (Kaba et al., 2023) in which one learns an energy over group elements and takes the minimizer as the canonical transformation. The latter has been further developed on symmetries defined by general Lie group actions (Shumaylov et al., 2025). Canonicalization has also been used for data alignment (Mondal et al., 2023; Schmidt & Stober, 2025) and for test-time optimization over transformations, where one searches over group actions to select a canonical representation before downstream inference (Singhal et al., 2025; Schmidt & Stober, 2024). A related line of work is frame averaging (Puny et al., 2020), which averages a network's output over a set of group transformations, and its extension to weighted frame averaging (Dym et al., 2024), where each datapoint is equipped with a probability distribution over the group and averaging is performed with respect to this measure, yielding continuity guarantees. In our work, we instead study canonicalization as a function of both the input and the network, and we establish continuity guarantees for symmetry-preserving continuous functions realized by our construction. Moreover, our approach is not restricted to symmetries defined via group actions, and allows working with more general augmentations for transforming datapoints. We refer to App. A for further discussion and comparison with related work.

3 Adaptive Canonicalization

In this section, we develop the general theory of adaptive canonicalization, and prove that it leads to continuous symmetry preserving networks with universal approximation properties.

3.1 Basic Definitions and Background

The function that maps each input x to the output $f(x)$ is denoted by $x \mapsto f(x)$. The free variable of a univariate function is denoted by $(\cdot)$, and by $(\cdot, \cdot\cdot)$ for a function of two variables. For example, the function $(x, y) \mapsto \sin(x)\exp(y)$ is also denoted by $\sin(\cdot)\exp(\cdot\cdot)$. We denote the infinity norm of $x = (x_d)_{d=1}^{D} \in \mathbb{R}^D$ by $|x| := \max_{1 \leq d \leq D} |x_d|$. We define the infinity norm of a continuous function $f : \mathcal{K} \to \mathbb{R}^D$ over a topological space $\mathcal{K}$ by $\|f\|_\infty = \sup_{x\in\mathcal{K}} |f(x)|$. If $\|f - y\|_\infty < \epsilon$ we say that y approximates f uniformly up to error ϵ. The set of all subsets of a set $\mathcal{K}$, i.e., the *power set*, is denoted by $2^{\mathcal{K}}$. When defining general metric spaces, we allow the distance between points to be ∞. This does not affect most of the common properties of metric spaces (see Burago et al. (2001)).

Function Spaces. We develop the definitions of adaptive canonicalization in general locally compact Hausdorff spaces. Two important examples of such a space is a compact metric space or $\mathbb{R}^J$.

Definition 1. *Let $\mathcal{K}$ be a locally compact Hausdorff space, and $D \in \mathbb{N}$.*

- *A function $f : \mathcal{K} \to \mathbb{R}^D$ is said to vanish at infinity if for every $\epsilon > 0$ there exists a compact set $K \subset \mathcal{K}$ such that $|f(x)| < \epsilon$ for every $x \in \mathcal{K} \setminus K$.*
- *The space of all continuous functions $f : \mathcal{K} \to \mathbb{R}^D$ that vanish at infinity, with the supremum norm $\|f\|_\infty = \max_{x\in\mathcal{K}} |f(x)|$ is denoted by $C_0(\mathcal{K}, \mathbb{R}^D)$.*

In adaptive canonicalization, we consider families of continuous functions where the $\epsilon - \delta$ formulation of continuity is uniform over the whole family, as defined next.

Definition 2. *Let $\mathcal{X}$ and $\mathcal{Y}$ be two metric spaces with metrics $d_\mathcal{X}$ and $d_\mathcal{Y}$ respectively. A family $\mathcal{F}$ of function $f : \mathcal{X} \to \mathcal{Y}$ is called* equicontinuous *if for every $x \in \mathcal{X}$ and every $\epsilon > 0$, there exists $\delta > 0$ such that every $z \in \mathcal{X}$ which satisfies $d_\mathcal{X}(x, z) < \delta$ also satisfies*

$$\forall f \in \mathcal{F} : \quad d_\mathcal{Y}(f(x), f(z)) < \epsilon.$$

Universal Approximation. *Universal approximation theorems (UAT)* state that any continuous function over some topological space can be approximated by a neural network. In such a case, the neural networks are said to be *universal approximators*, as defined next.

Definition 3. *Let $\mathcal{K}$ be a locally compact Hausdorff space and $D \in \mathbb{N}$. A set of continuous functions $\mathcal{N}(\mathcal{K}, \mathbb{R}^D) \subset C_0(\mathcal{K}, \mathbb{R}^D)$ is said to be a* universal approximator *of $C_0(\mathcal{K}, \mathbb{R}^D)$ if for every $f \in C_0(\mathcal{K}, \mathbb{R}^D)$ and $\epsilon > 0$ there is a function $\theta \in \mathcal{N}(\mathcal{K}, \mathbb{R}^D)$ such that*

$$\forall x \in \mathcal{K} : \quad |f(x) - \theta(x)| < \epsilon.$$

In the above definition, we interpret $\mathcal{N}(\mathcal{K}, \mathbb{R}^D)$ as a space of neural networks. A UAT is hence any theorem which shows that some set of neural networks is a universal approximator. Two examples of UATs are: 1) multilayer perceptrons (MLP) are universal approximators of $C_0(\mathcal{K}, \mathbb{R}^D)$ for compact subset $\mathcal{K}$ of the Euclidean space $\mathbb{R}^D$ (Hornik et al., 1989; Cybenko, 1989), and 2) DeepSets (Zaheer et al., 2017) are universal approximators of continuous functions from multi-sets to $\mathbb{R}^d$. See App. B for more details.

3.2 Adaptive Canonicalization

In the general setting of adaptive canonicalization, we have a domain of inputs $\mathcal{G}$ which need not have any structure apart for being a set, e.g., the set of graphs. We consider continuous functions $f : \mathcal{K} \to \mathbb{R}^L$ over a "nice" domain $\mathcal{K}$, e.g., $\mathcal{K} = \mathbb{R}^J$. Such functions can be approximated by neural networks. We then pull-back f to be a function from $\mathcal{G}$ to $\mathbb{R}^L$ using a mapping $\rho_f : \mathcal{G} \mapsto \mathcal{K}$ that depends on (is adapted to) f. Namely, we consider $f(\rho_f(\cdot)) : \mathcal{G} \to \mathbb{R}^D$. The following definitions assure that such a setting leads to functions with nice properties, as we show in subsequent sections.

Definition 4. *Let $\mathcal{K}$ be a locally compact Hausdorff space, $\mathcal{G}$ be a set, and $D \in \mathbb{N}$. A mapping $\rho = \rho_{(\cdot)}(\cdot\cdot) : C_0(\mathcal{K}, \mathbb{R}^D) \times \mathcal{G} \to \mathcal{K}$, $(f, g) \mapsto \rho_f(g)$, is called an* adaptive canonicalization *if the set of functions*

$$\{f \mapsto f \circ \rho_f(g) \mid g \in \mathcal{G}\}$$

is equicontinuous (as functions $C_0(\mathcal{K}, \mathbb{R}) \to \mathbb{R}^D$). Here, $f \circ \rho_f(g) := f(\rho_f(g))$.

Next, we define the function space that we would like to approximate using adaptive canonicalization.

Definition 5. *Let $\mathcal{K}$ be a locally compact Hausdorff space, $\mathcal{G}$ be a set, and $D \in \mathbb{N}$. Let ρ be an adaptive canonicalization, and let $f \in C_0(\mathcal{K}, \mathbb{R}^D)$. The function*

$$f \circ \rho_f : \mathcal{G} \to \mathbb{R}^D, \quad g \mapsto f\big(\rho_f(g)\big)$$

is called an adaptive canonicalized continuous function*, or a* canonicalized function *in short.*

In Sec. 3.4, we show that for an important class of adaptive canonicalizations the set of adaptive canonicalized continuous functions is exactly the set of all symmetry preserving continuous functions.

It is now direct to prove the following universal approximation theorem.

Theorem 6 (Universal approximation of adaptive canonicalized functions)**.** *Let $\mathcal{N}(\mathcal{K}, \mathbb{R}^D)$ be a universal approximator of $C_0(\mathcal{K}, \mathbb{R}^D)$, and $f \circ \rho_f$ an adaptive canonicalized continuous function. Then, for every $\epsilon > 0$, there exists a network $\theta \in \mathcal{N}(\mathcal{K}, \mathbb{R}^D)$ such that for every $g \in \mathcal{G}$*

$$|f \circ \rho_f(g) - \theta \circ \rho_\theta(g)| < \epsilon.$$

Proof. Let $\epsilon > 0$. By Def. 4, there exists $\delta > 0$ such that

$$\forall y \in C_0(\mathcal{K}, \mathbb{R}^D) : \ \|f - y\|_\infty < \delta \Rightarrow \Big(\forall g \in \mathcal{G} : \ |f \circ \rho_f(g) - y \circ \rho_y(g)| < \epsilon\Big). \tag{1}$$

By the the universal approximation property, there exists a network θ such that $\|f - \theta\|_\infty < \delta$. Hence, by (1), $\quad \forall g \in \mathcal{G} : \ |f \circ \rho_f(g) - \theta \circ \rho_\theta(g)| < \epsilon$. □

3.3 Prior Maximization Adaptive Canonicalization

Prior maximization is a special case of adaptive canonicalization, where ρ_f is chosen to maximize some prior on the output of f. The maximization is done over a space of transformations $\kappa_u : \mathcal{G} \to \mathcal{K}$ parameteried by u, i.e., maximizing the prior of $f(\kappa_u(g))$ w.r.t. u.

Let $\mathcal{U}$ be metric space, and for every $u \in \mathcal{U}$, let

$$\kappa_{(\cdot)}(\cdot\cdot) : \mathcal{U} \times \mathcal{G} \to \mathcal{K}, \quad (u, g) \mapsto \kappa_u(g) \in \mathcal{K}.$$

Suppose that $\kappa_u(g)$ is continuous in u for every $g \in \mathcal{G}$. We call κ a *transformation family*, transforming objects in $\mathcal{G}$ into points in $\mathcal{K}$, where different $u \in \mathcal{U}$ define different transformations. Let $H = (h_1, \ldots, h_D)$, where $h_d : \mathbb{R} \to \mathbb{R}$ for each d, be a sequence of continuous monotonic functions, that we call the ensemble of *priors*. We call each h_d a *prior*. We denote $H \circ f := (h_d \circ f_d)_{d=1}^D$.

For every $f = (f_1, \ldots, f_D) \in C_0(\mathcal{K}, \mathbb{R}^D)$, $g \in \mathcal{G}$ and d, assume that $h_d \circ f_d(\kappa_{(\cdot)}(g))$ attains a maximum in $\mathcal{U}$. This is the case for example when $\mathcal{U}$ is compact. Define

$$\rho_f(g) = \big(\rho^d_{f_d}(g)\big)_{d=1}^D := \Big(\big\{\kappa_{u_*}(g) \,\big|\, h_d \circ f_d\big(\kappa_{u_*}(g)\big) = \max_{u \in \mathcal{U}} h_d \circ f_d\big(\kappa_u(g)\big)\big\}\Big)_{d=1}^D \in \big(2^{\mathcal{K}}\big)^D. \tag{2}$$

Note that $\rho_f : \mathcal{G} \to (2^{\mathcal{K}})^D$. By abuse of notation, we also denote by ρ_f the mapping that returns some arbitrary sequence of points $(x_d \in \rho^d_{f_d}(g))_{d=1}^D \in \mathcal{K}^D$ for each $g \in \mathcal{G}$. The choice of the specific point in $\rho^d_{f_d}(g)$ does not affect the analysis. We interpret ρ_f as a function that takes an input g and canonicalize it separately with respect to each output channel f_d, adaptively to f_d.

When used for classification, we interpret each output channel $f_d \circ \rho^d_{f_d}(g) \in [0, 1]$ as a binary classifier, i.e., representing the probability of g being in class d vs. not being in class d. This multiclass classification setting is called *one vs. rest*, where a standard loss is a sum of D binary cross-entropies (Rifkin & Klautau, 2004; Galar et al., 2011; Allwein et al., 2000).

Definition 7. *Consider the above setting. The mapping ρ defined by (2) is called* prior maximization. *If in addition $\mathcal{G}$ has a metric such that for every $f \in C_0(\mathcal{K}, \mathbb{R}^D)$ the family $\{g \mapsto f(\kappa_u(g))\}_{u \in \mathcal{U}}$ is equicontinuous, ρ is called* continuous prior maximization.

In Thm. 8 we show that prior maximization is indeed adaptive canonicalization.

Note that the condition of $g \mapsto f(\kappa_u(g))$ being equicontinuous is satisfied for well known settings of equivariant machine learning. For example, let $\mathcal{U} = \mathcal{SO}(3)$ be the space of 3D rotations, and $\mathcal{G} = \mathcal{K} = \mathcal{B}^N$ the set of sequences of N points in the 3D unit ball $\mathcal{B}$, i.e., the space of point clouds. We consider the rotation $g \mapsto \kappa_u(g)$ of the point cloud g by $u \in \mathcal{U}$. Since $\mathcal{G}$ and $\mathcal{U}$ are compact metric spaces, and $(g, u) \mapsto \kappa_u(g)$ is continuous, κ and f must be uniformly continuous. Hence, $\{g \mapsto f(\kappa_u(g))\}_{u \in \mathcal{U}}$ is equicontinuous. In fact, whenever $\mathcal{G}$ is compact and κ continuous w.r.t. (u, g), it is automatically also uniformly continuous, so ρ is a continuous prior maximization. See App. C for additional examples of continuous prior maximization.

Properties of Prior Maximization.

Theorem 8. *In prior maximization, each $\rho^d : C_0(\mathcal{K}, \mathbb{R}) \times \mathcal{G} \to \mathcal{K}$ is adaptive canonicalization.*

Proof. Consider without loss of generality the case where the output dimension is $D = 1$. Since the specific values of $H = h_1$ do not matter, only if it is ascending or descending, without loss of generality suppose $H(x) = x$, in which case prior maximization maximizes directly the output of $f \circ \kappa_u(g)$ with respect to u. Consider an arbitrary maximizer $\rho_f(g) \in \arg\max_{u \in \mathcal{U}} f(\kappa_u(g))$ for each $f \in C_0(\mathcal{K}, \mathbb{R})$. The choice of the maximizer does not affect the analysis. Now, if $f, y \in C_0(\mathcal{K}, \mathbb{R})$ satisfy $\|f - y\|_\infty < \epsilon$, then also for every $u \in \mathcal{U}$, $|f(\kappa_u(g)) - y(\kappa_u(g))| < \epsilon$. Let $u_0 \in \mathcal{U}$ be a maximizer of $f(\kappa_u(g))$. We have $y(\kappa_{u_0}(g)) > f(\kappa_{u_0}(g)) - \epsilon$, so

$$\max_u y(\kappa_u(g)) > \max_u f(\kappa_u(g)) - \epsilon.$$

Similarly, we have $\max_u f(\kappa_u(g)) > \max_u y(\kappa_u(g)) - \epsilon$. Together,

$$\left|\max_{u\in\mathcal{U}} f\circ\kappa_u(g) - \max_{u\in\mathcal{U}} y\circ\kappa_u(g)\right| < \epsilon. \tag{3}$$

Hence, $f \mapsto f\circ\rho_f(g)$ is Lipschitz continuous with Lipschitz constant 1 for every $g\in\mathcal{G}$, and therefore equicontinuous over the parameter $g\in\mathcal{G}$. □

This immediately gives a universal approximation theorem for prior maximization as a corollary of Thm. 6. Moreover, we can show that continuous prior maximization gives functions continuous in $\mathcal{G}$. This is one of the main distinctions between prior maximization and standard canonicalization.

Theorem 9. *Consider a continuous prior maximization ρ (Def. 7). Then, $f\circ\rho_f$ is continuous.*

Proof. Let $\epsilon > 0$. For every $g\in\mathcal{G}$ there is $\delta = \delta_{\epsilon,g} > 0$ such that for every $g'\in\mathcal{G}$ with $d(g,g') < \delta$ and every $u\in\mathcal{U}$ we have $|f(\kappa_u(g)) - f(\kappa_u(g'))| < \epsilon$. Now, by the same argument as in (3),

$$\left|\max_u f(\kappa_u(g)) - \max_u f(\kappa_u(g'))\right| < \epsilon.$$

□

3.4 Symmetry Preserving Prior Maximization

Consider the following additional assumptions on the construction of continuous prior maximization. Suppose that the space $\mathcal{G}$ is a disjoint union of metric spaces $\mathcal{G}_j$ with finite distances, i.e., $\mathcal{G} = \uplus_j\mathcal{G}_j$. Here, j may run on a finite or infinite index set. We define the metric d in $\mathcal{G}$ as follows. For $g_j\in\mathcal{G}_j$ and $g_i\in\mathcal{G}_i$, $d(g_j,g_i) = \infty$ if $j\neq i$ and $d(g_j,g_i) = d_j(g_j,g_i) < \infty$ if $i = j$, where d_j is the metric in $\mathcal{G}_j$. In the theory of metric spaces, the spaces $\mathcal{G}_j$ are called *galaxies* of $\mathcal{G}$. This construction is useful for data which does not have a uniform notion of dimension, e.g., graphs. For example, each galaxy in this case can be the space of adjacency matrices of a fixed dimension with a standard matrix distance.

For each j, let $\mathcal{U}_j$ be a group acting continuously on $\mathcal{G}_j$ by $\pi_j(u_j)g_j$. Namely, $\pi_j(u_j):\mathcal{G}_j\to\mathcal{G}_j$ is continuous for every $u_j\in\mathcal{U}_j$, and for every $u_j'\in\mathcal{U}_j$ and $g_j\in\mathcal{G}_j$ we have $\pi_j(u_j')\pi_j(u_j)g_j = \pi_j(u_j'u_j)g_j$ and $\pi_j(e_j)g_j = g_j$, where e_j is the identity of $\mathcal{G}_j$. Define $\mathcal{U} = \uplus_j\mathcal{U}_j$. Namely, $\mathcal{U}$ is the metric space with galaxies $\mathcal{U}_j$ similarly to the construction of $\mathcal{G}$. Let π be a mapping that we formally call an action of $\mathcal{U}$ on $\mathcal{G}$, defined for $u = u_i\in\mathcal{U}_i\subset\mathcal{U}$ and $g = g_j\in\mathcal{G}_j\subset\mathcal{G}$ by $\pi(u)g = \pi_i(u_i)(g_j)$ if $i = j$ and $\pi(u)g = g$ if $i\neq j$.

Definition 10. *Consider the above setting and a continuous prior maximization ρ. Let $P:\mathcal{G}\to\mathcal{K}$ be continuous, and suppose that the transformation family κ is of the form $\kappa_u = P\circ\pi(u)$. We call κ a* symmetry preserving transformation family*, and ρ a* symmetry preserving prior maximization*.*

Note that whenever the spaces $\mathcal{U}_j$ are compact, the functions $u\mapsto h_d\circ f_d\big(\kappa_u(g)\big)$ of (2) are guaranteed to attain a maximum, even though the space $\mathcal{U} = \uplus_j\mathcal{U}_j$ is in general not compact. Hence, the above setting with compact $\mathcal{U}_j$ is an example of prior maximization. More generally, if the restriction of the setting to $\mathcal{U}_j$ and $\mathcal{G}_j$ is (continuous) prior maximization for a single j, then the setting for $\mathcal{U}$ and $\mathcal{G}$ is also a (continuous) prior maximization.

Definition 11. *We call a function $Q:\mathcal{G}\to\mathbb{R}^D$* continuous symmetry preserving *if there exists $F:\mathcal{K}\to\mathbb{R}^D$ in $C_0(\mathcal{K};\mathbb{R}^D)$ such that for all $u\in\mathcal{U}$ and $g\in\mathcal{G}$, $\quad Q(g) = F(P(\pi(u)(g)))$.*

When $\mathcal{K} = \mathcal{G} = \mathcal{G}_1$ and P is the identity, a symmetry preserving continuous function is a continuous function which is invariant to the action of u, i.e., the classical case in equivariant machine learning.

Properties of Symmetry Preserving Prior Maximization. We already know by Thm. 9 that $f\circ\rho_f$ is a continuous function when ρ is a symmetry preserving prior maximization. We next show that the set of functions of the form $f\circ\rho_f$ exhaust the space of all continuous symmetry preserving functions.

Theorem 12. *Let ρ be a symmetry preserving adaptive canonicalization. Then,*

1. *Any continuous symmetry preserving function can be written as $f\circ\rho_f$ for some $f\in C_0(\mathcal{K},\mathbb{R}^D)$.*
2. *For any $f\in C_0(\mathcal{K};\mathbb{R}^D)$, the function $f\circ\rho_f:\mathcal{G}\to\mathbb{R}^D$ is continuous symmetry preserving.*

Proof. Without loss of generality, consider the case $D = 1$ and $H(x) = x$.

1. For $Q(g) = F(P(\pi(u)(g)))$, take $f = F$. Then, by definition of symmetry preservation, for every $u \in \mathcal{U}$: $\quad f \circ \rho_f(g) = \max_v F(P \circ \pi(v)g) = F(P \circ \pi(u)g) = Q(g)$.
2. For any $u = u_i \in \mathcal{U}_i$, since $\pi_i(u_i)$ is an action, for any $g = g_j \in \mathcal{G}_j$,
$$f \circ \rho_f(\pi(u)g) = \max_{v_j \in \mathcal{U}_j} f(P \circ \pi_j(v_j)\pi_i(u_i)g_j) = \max_{v_j \in \mathcal{U}_j} f(P \circ \pi_j(v_j)g_j) = f \circ \rho_f(g).$$

□

This leads to the following UAT.

Theorem 13. *Consider a symmetry preserving prior maximization and let $\mathcal{N}(\mathcal{K}, \mathbb{R}^D)$ be a universal approximator of $C_0(\mathcal{K}, \mathbb{R}^D)$. Then, any continuous symmetry preserving function can be approximated uniformly by $\theta \circ \rho_\theta$ for some network $\theta \in \mathcal{N}(\mathcal{K}, \mathbb{R}^D)$.*

4 Application of Adaptive Canonicalization to Anisotropic Geometric Networks

In this section, we propose two architectures based on adaptive canonicalization which can be interpreted as anisotropic. First, a spectral graph neural network (GNN) which is sensitive to directionality within eigenspaces. Then, a 3D point cloud network which is sensitive to 3D directions.

Basic Notations for Graphs and Vectors. We denote by $\mathbb{N}_0$ the set of nonnegative integers. We denote matrices and 2D arrays by boldface capital letters, e.g., $\mathbf{B} \in \mathbb{R}^{N \times T}$. We denote $[N] = \{1, \ldots, N\}$ for $N \in \mathbb{N}$. We denote by $\mathbf{B}(:, j)$ and $\mathbf{B}(j, :)$ the j'th column and row of the matrix $\mathbf{B}$ respectively. A graph is denoted by $G = ([N], \mathbf{A}, \mathbf{S})$, where $[N]$ is the set of N vertices, $\mathbf{A} \in \mathbb{R}^{N \times N}$ is the adjacency matrix, and $\mathbf{S} \in \mathbb{R}^{N \times T}$ is an array representing the signal. Namely, $\mathbf{S}(n, :) \in \mathbb{R}^T$ is the feature at node n. A graph shift operator (GSO) is a self-adjoint operator that respects in some sense the graph's connectivity, e.g., a graph Laplacian or the adjacency matrix.

4.1 Anisotropic nonlinear spectral filters

Consider graphs with a self-adjoint GSO $\mathcal{L}$ (e.g., a graph Laplacian or the adjacency matrix) and T-channel signals (over the nodes). The number of nodes N varies between graphs. Consider predefined bands $b_0 < b_1 < \ldots < b_B \in \mathbb{R}$, and their indicator functions $P_k := \mathbb{1}_{[b_{k-1}, b_k)} : \mathbb{R} \to \mathbb{R}_+$[1]. For each $k \in [B]$, consider the space of signals $\mathcal{X}_k$ in each band $[b_{k-1}, b_k)$, namely, the range of the orthogonal projection $P_k(\mathcal{L})$. Let M_k be the dimension of $\mathcal{X}_k$. We also call $\mathcal{X}_k$ the k'th band. See App. D for details on how to plug operators into functions via functional calculus. Consider an orthogonal basis $\mathbf{X}_k = (\mathbf{X}_k(:, j))_j \in \mathbb{R}^{N \times M_k}$ for each band $\mathcal{X}_k$. In this setting, the symmetry is the choice of the orthogonal basis within each band $\mathcal{X}_k$. Namely, for $\mathbf{X}_k \in \mathbb{R}^{N \times M_k}$ and orthogonal matrix $\mathbf{U}_k \in \mathbb{R}^{M_k \times M_k}$, the bases $\mathbf{X}_k$ and $\mathbf{X}_k\mathbf{U}_k$ are treated as different vectorial representations of the same linear space $\mathcal{X}_k$.

In the above setting, the galaxies $\mathcal{G}_j$ are defined as follows. Each galaxy is indexed by $j = (N, \mathbf{M}) := (N; M_1, \ldots, M_B)$ where $N \in \mathbb{N}$ is the size of the graph, and $M_k \in \mathbb{N}_0$ for $k \in [B]$ are the dimensions of the bands, satisfying $\sum_k M_k \leq N$. Each $\mathcal{G}_{N,\mathbf{M}}$ is the space of pairs of a signal $\mathbf{S} \in \mathbb{R}^{N \times T}$ and a sequence of K orthogonal matrices of height N and widths $M_1, \ldots, M_B$. We denote the matrix with dimension 0 by $\emptyset$. For two signals $\mathbf{S}, \mathbf{Q}$ and orthogonal bases sequences $\mathbf{X}, \mathbf{Y}$, the metric $d\big((\mathbf{X}, \mathbf{S}), (\mathbf{Y}, \mathbf{Q})\big)$ is defined to satisfy

$$d(\mathbf{X}, \mathbf{Y})^2 = \|\mathbf{S} - \mathbf{Q}\|_2^2 + \sum_{k=1}^{B} \sum_{j=1}^{M_k} \|\mathbf{X}_k[:, j] - \mathbf{Y}_k[:, j]\|_2^2 = \|\mathbf{S} - \mathbf{Q}\|_2^2 + \|\mathbf{X} - \mathbf{Y}\|_2^2,$$

where for $\mathbf{z} = (z_n)_n$, $\|\mathbf{z}\|_2^2 = \sum_n z_n^2$. By definition, the distance between any two points from two different galaxies is ∞.

The transfomation family $\kappa_{\mathbf{U}}$ apply unitary operators $\mathbf{U}_k \in \mathcal{O}(M_k)$ form the right on each $\mathbf{X}(:, k)$, where $\mathcal{O}(M_k)$ is the group of orthogonal matrices in $\mathbb{R}^{M_k \times M_k}$. Namely, $\mathcal{U}_{N;\mathbf{M}} := \prod_{k=1}^{B} \mathcal{O}(M_k)$.

[1] $\mathbb{1}_{[b_{k-1}, b_k)}(x)$ is the function that returns 1 if $x \in [b_{k-1}, b_k)$ and 0 otherwise.

We now define the operator C that computes the spectral coefficients of the signal $\mathbf{S}$ with respect to the spectral basis $\mathbf{X}$. Namely,

$$C(\mathbf{X},\mathbf{S}) = \big(C_k(\mathbf{X},\mathbf{S})\big)_{k=1}^{B} := (\mathbf{X}(:,k)^\top \mathbf{S} \in \mathbb{R}^{M_k \times T})_{k=1}^{B}.$$

In *anisotropic nonlinear spectral filters (A-NLSF)*, we consider a symmetryless neural network Ψ that operates on the space of spectral coefficients of the signal, e.g., a multilayer perceptron (MLP). To define Ψ consistently, over a Euclidean space of fixed dimension, we extend or truncate the sequence of spectral coefficients as follows. Let $J_1, \dots, J_B$ be a predefined sequence of integers, and denote $J = \sum_k J_k$. We define P as the mapping that takes $(\mathbf{X},\mathbf{S})$ as input, first compute the spectral coefficients $\big(C_k(\mathbf{X},\mathbf{S})\big)_{k=1}^{B}$, and then truncates or pads with zeros each $C_k(\mathbf{X},\mathbf{S})$ to be in $\mathbb{R}^{J_k \times T}$. Namely, $P(\mathbf{X},\mathbf{S}) = \big(P_k(\mathbf{X},\mathbf{S}) \in \mathbb{R}^{J_k \times T}\big)_{k=1}^{B} \in \mathbb{R}^{J \times T}$. Here, the matrix $\emptyset$ is padded to the zero matrix $\mathbf{0} \in \mathbb{R}^{J_k \times T}$. Hence, the symmetryless network Ψ maps $\mathbb{R}^{J \times T}$ to $\mathbb{R}^{D}$. A more detailed derivation of the construction is given in App. E.3.

We call this architecture anisotropic for the following reason. Consider for example the grid graph with $N \times N$ vertices. Since spectral filters are based on functional calculus (see App. D), they are invariant to graph automorphism, and hence to rotations. This means that spectral filters treat the x and y axes equally, and any filter is isotropic in the spatial domain. On the other hand, our symmetryless network Ψ can operate differently on the x and y axes, and we can implement general directional filters on images with Ψ. In the general case, Ψ can operate differently on Fourier modes from the same eigenspace, which we interpret as directions within the eigenspace, while standard GNNs cannot. See App. G.2 for more details.

4.2 Anisotropic Point Cloud Networks

Here, we present a point cloud network which is a combination of an equivariant network with adaptive canonicalization. Namely, we consider a permutation invariant network Ψ like DeepSet (Zaheer et al., 2017) or DGCNN (Wang et al., 2019), and to attain 3D rotation invariance in addition we incorporate prior maximization. Together, the method is invariant both to permutations and rotations. We call this method anisotropic since Ψ does not respect the rotation symmetries, and is hence sensitive to directions in the x, y, z space.

We restrict the analysis to multi-sets of a fixed number of points N. Multisets are sets where repetitions of elements are allowed. Here, we formally define a multi-set as an equivalence class of arrays up to permutation. To define this, let $\mathcal{S}_N$ be the symmetric group of N elements, i.e., the group of permutations. given $s \in \mathcal{S}$ and $\mathbf{X} \in \mathbb{R}^{N \times J}$, let $\rho(s)\mathbf{X}$ be the permutation that changes the order of the rows $\mathbf{X}$ according to s. We say that $\mathbf{X} \sim \mathbf{Y}$ if there is $s \in \mathcal{S}_N$ such that $\mathbf{X} = \rho(s)\mathbf{Y}$. The equivalence class $[\mathbf{X}]$ is defined as $\{\mathbf{Y} \in \mathbb{R}^{N \times J} \mid \mathbf{Y} \sim \mathbf{X}\}$, and the space of equivalence classes, also called the *quotient space*, is denoted by $(\mathbb{R}^{N \times J}/\sim) := \{[\mathbf{X}] \mid \mathbf{X} \in \mathbb{R}^{N \times J}\}$. We identify the space $(\mathbb{R}^{N \times J}/\sim)$ with the space of multisets.

In App. B.2 we show that the quotient space has a natural metric. We hence take $\mathcal{G} = \mathcal{K}$ consisting of a single galaxy $\mathcal{G}_N = (\mathbb{R}^{N \times J}/\sim)$. We moreover show in App. B.2 that any universal approximator of permutation invariant functions in $C_0(\mathbb{R}^{N \times J}, \mathbb{R}^{D})$, e.g., DeepSet (Zaheer et al., 2017), canonically gives a universal approximator of general continuous functions in $C_0(\mathbb{R}^{N \times J}/\sim, \mathbb{R}^{D})$.

The symmetry in our adaptive canonicalization is 3D rotations $\pi(u) \in \mathbb{R}^{3 \times 3}$, where u is in the rotation group $\mathcal{SO}(3)$. Namely, we consider $J = 3$, and rotated the rows of $\mathbf{X} \in \mathbb{R}^{N \times 3}$ via $\mathbf{X}\pi(u^{-1})$. We take P as the identity. Note that this construction can be easily extended to multisets of arbitrary sizes, by considering the galaxies $\mathcal{G}_N = \mathbb{R}^{N \times 3}/\sim$ and groups $\mathcal{U}_N = \mathcal{S}_N$ for all $N \in \mathbb{N}$. For details on the theoretical construction see App. B.2, and for details on the architecture see App. E.4.

4.3 Additional Applications of Adaptive Canonicaluzation

Our adaptive canonicalization is a general framework and is not limited to the two applications considered above. For example, it can be instantiated for image truncation with a pretrained network, where the symmetry corresponds to different crops and prior maximization selects the crop on which the model is most confident (see App. E.5). Note that our setting and theoretical results also apply to pretrained networks on downstream tasks. Our formulation also accommodates several instances

Table 1: Classification performance on grid signal orientation task and graph classification benchmarks from TUDataset. The highest accuracy in 🟦 and the second highest in 🟥.

	Toy Example	TUDataset				
		MUTAG	PTC	ENZYMES	PROTEINS	NCI1
MLP	50.03±0.1	79.31±3.5	63.98±2.0	42.17±2.8	75.08±3.4	77.34±1.6
GCN	50.01±0.1	81.63±3.1	60.22±1.9	43.66±3.4	75.17±3.7	76.29±1.8
GAT	49.95±0.0	83.17±4.4	62.31±1.4	39.83±3.7	74.72±4.1	74.01±4.3
GIN	50.00±0.1	83.29±3.6	63.25±2.3	45.69±2.6	76.02±2.9	79.84±1.2
ChebNet	50.12±0.1	82.15±1.6	64.06±1.2	50.42±1.4	74.28±0.9	76.98±0.7
FA+GIN	49.99±0.1	84.07±2.4	66.58±1.8	52.64±2.2	79.53±2.5	80.23±0.9
OAP+GIN	50.03±0.0	84.95±2.0	67.35±1.1	58.40±1.6	83.41±1.4	80.97±1.1
NLSF	50.07±0.1	84.13±1.5	68.17±1.0	65.94±1.6	82.69±1.9	80.51±1.2
S^{2}GNN	49.93±0.1	82.70±2.1	67.34±1.5	63.26±2.8	78.52±1.9	75.62±2.0
A-NLSF	99.38±0.2	87.94±0.9	73.16±1.2	73.01±0.8	85.47±0.6	82.01±0.9

of continuous prior maximization, e.g., unbounded point clouds under rotations and continuous-to-discrete image settings with rotations and other image transformations (see App. C). Finally, to broadening the applicability of our approach beyond classification, we explore using the adaptive canonicalization mechanism for point cloud segmentation, further (see App. G.10).

5 Experiments

We evaluate the anisotropic nonlinear spectral filters (Sec. 4.1) on toy problems and graph classification, and test the anisotropic point cloud network (Sec. 4.2) on point cloud classification. The experimental details, including experimental setups and hyperparameters, are in App. F. Additional experiments, e.g., ablation study, are in App. G.

Maximization method. We approximate the prior maximization by sampling a finite set of transformations from a probability measure on the transformation space, evaluating the prior for all candidates, and retaining the transformation that yields the largest prior value with the one-vs-rest classification objective. We then refine the selected candidate locally by running a few steps of gradient descent from the best sampled transformation (see App. E.2).

5.1 Experimental Evaluation of Anisotropic Nonlinear Spectral Filters

Illustrative Toy Problems: Grid Signal Orientation Task. To showcase the effectiveness of A-NLSF, we study a toy classification task on a grid-split channel orientation. We consider a square grid on the torus, and each node has two channels. In addition, the grid is further partitioned vertically into two equal disjoint halves. Channel 1 is nonzero only on the left half, and Channel 2 is only on the right half. In class 0, both channels are 1-frequency along x, and in class 1, Channel 1 is 1-frequency along x and Channel 2 is 1-frequency along y. The task is to decide if the frequency at the two channels is in the same orientation. See App. F for further details.

We compare A-NLSF with the following baselines: (i) MLP, (ii) GCN (Kipf, 2016), (iii) GAT (Veličković et al., 2017), (iv) GIN (Xu et al., 2018), (v) ChebNet (Defferrard et al., 2016a), (vi) NLSF (Lin et al., 2024a), and (vii) S^{2}GNN (Geisler et al., 2024). In addition, we test canonicalization baselines by combining GIN with frame averaging (FA) (Puny et al., 2021) and orthogonalized axis projection (OAP) (Ma et al., 2024). Tab. 1 reports the classification results. We see that competing methods remain at chance level while A-NLSF achieves high accuracy by adaptively resolving ambiguities, showing the advantage for orientation-sensitive learning on disjoint supports.

Graph Classification on TUDataset. We further evaluate A-NLSF on graph classification benchmarks from TUDataset (Morris et al., 2020): MUTAG, PTC, ENZYMES, PROTEINS, and NCI1, and follow the experimental setup (Ma et al., 2019; Ying et al., 2018; Zhang et al., 2019b) (see App. F). We compare with the same baselines as in the grid signal orientation tasks. Tab. 1 summarizes the classification performance. Canonicalization baselines generally improve over GIN. Notably, we observe that A-NLSF outperforms competing baselines, suggesting that our AC provides more informative representations compared to a fixed, precomputed canonical form or isotropic filters.

Molecular Classification on OGB Datasets. To further assess the effectiveness of A-NLSF, we evaluate on large-scale molecular and protein benchmarks from Open Graph Benchmark (OGB) (Hu et al., 2020): ogbg-molhiv, ogbg-molpcba, and ogbg-ppa. We compare with GCN, GIN, GatedGCN (Bresson & Laurent, 2017), PNA (Corso et al., 2020), GraphTrans (Wu et al., 2021), SAT (Chen et al., 2022), GPS (Rampášek et al., 2022), SAN (Kreuzer et al., 2021), and the canonicalization method OAP+GatedGCN. The molecular classification results are reported in Tab. 2. We see that our method achieves consistent improvements across these datasets, leading to improved generalization in classification.

Table 2: Molecular and protein classification performance on OGB datasets.

	ogbg-molhiv	ogbg-molpcba	ogbg-ppa
	AUROC ↑	Avg. Precision ↑	Accuracy ↑
GCN	0.7599±0.0119	0.2424±0.0034	0.6857±0.0061
GIN	0.7707±0.0149	0.2703±0.0023	0.7037±0.0107
GatedGCN	0.7687±0.0136	0.2670±0.0020	0.7531±0.0083
PNA	0.7905±0.0132	0.2838±0.0035	-
GraphTrans	-	0.2761±0.0029	-
SAT	-	-	0.7522±0.0056
GPS	0.7880±0.0101	0.2907±0.0028	0.8015±0.0033
SAN	0.7785±0.2470	0.2765±0.0042	-
OAP+GatedGCN	0.7802±0.0128	0.2783±0.0024	0.7745±0.0098
A-NLSF	0.8019±0.0152	0.2968±0.0022	0.8149±0.0067

5.2 Experimental Evaluation of Anisotropic Point Cloud Networks

To evaluate adaptive canonicalization on point cloud classification, we test ModelNet40 (Wu et al., 2015). The dataset consists of 12,311 shapes from 40 categories, with 9,843 samples for training and 2,468 for testing. We build on two point cloud architectures, PointNet (Qi et al., 2017a) and DGCNN (Wang et al., 2019), and apply adaptive canonicalization into their pipeline, denoted respectively AC-PointNet and AC-DGCNN (see Sec. 4.2 and App. E.4). Following Esteves et al. (2018); Deng et al. (2021), we perform on-the-fly rotation augmentation during training, where the dataset size remains unchanged, and for the test set, each test example is arbitrarily rotated. We compare with PointNet, DGCNN, equivariant networks VN-PointNet and VN-DGCNN from Deng et al. (2021), canonicalization methods CN-PointNet and CN-DGCNN from Kaba et al. (2023), and traditional augmentation baselines where the training set is statically expanded with pre-generated rotated samples, denoted PointNet-Aug and DGCNN-Aug. For further details, see App. F. Tab. 3 shows the classification performance. We observe that our method outperforms the competing baselines and it allows the model to learn both local geometric features and optimal reference alignments for each class.

Table 3: Classification results on ModelNet40. Results of competing methods marked with * are taken from Deng et al. (2021); Luo et al. (2022); Kaba et al. (2023).

	Accuracy
PointNet	74.7*
DGCNN	88.6*
PointNet-Aug	75.8±0.9
DGCNN-Aug	89.0±1.0
VN-PointNet	77.2*
VN-DGCNN	90.2*
CN-PointNet	79.7±1.3 *
CN-DGCNN	90.0±1.1 *
AC-PointNet	81.1±0.7
AC-DGCNN	91.6±0.6

6 Conclusions

We introduce adaptive canonicalization based on prior maximization, a general framework for equivariant machine learning in which the standard form depends on both the input and the network. We prove that our method is continuous, symmetry preserving, and has universal approximation properties. We demonstrate the applicability of our theory in two settings: resolving eigenbasis ambiguities in spectral graph neural networks, and handling rotational symmetries in point clouds.

Limitations and Future Work. Our framework is naturally suited to classification tasks, and at the current scope of the paper, we did not address regression tasks. We will extend the adaptive canonicalization to regression in future work. Another limitation of our approach is that prior maximization requires solving D optimizations at runtime for the D classes. In future work, we will reduce this to a single optimization to improve efficiency.

Ethics statement

This work introduces a novel method for handling symmetry for equivariant machine learning, with a focus on theory and with application to spectral graph neural networks and point cloud networks. The experiments are conducted using simulated toy problems and public datasets, and therefore there is no concerns related to privacy, consent, or potential harm to living subjects. As the data employed are technical and free from sensitive or identifiable content, the research does not raise any apparent ethical concerns. Accordingly, no additional ethical approval was required for this study.

Reproducibility Statement

For the theoretical results, we include the main proofs in the paper and present additional analysis and illustrative examples in App. B and App. C. For the empirical study, the implementation details are reported in App. F. The source code will be released on GitHub upon publication.

References

P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2008.

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2623–2631, 2019.

Erin L Allwein, Robert E Schapire, and Yoram Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of machine learning research*, 1(Dec):113–141, 2000.

Brandon Anderson, Truong Son Hy, and Risi Kondor. Cormorant: Covariant molecular neural networks. *Advances in neural information processing systems*, 32, 2019.

Matan Atzmon, Koki Nagano, Sanja Fidler, Sameh Khamis, and Yaron Lipman. Frame averaging for equivariant shape space learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 631–641, 2022.

László Babai and Eugene M Luks. Canonical labeling of graphs. In *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, pp. 171–183, 1983.

Sourya Basu, Pulkit Katdare, Prasanna Sattigeri, Vijil Chenthamarakshan, Katherine Driggs-Campbell, Payel Das, and Lav R Varshney. Efficient equivariant transfer learning from pretrained models. *Advances in Neural Information Processing Systems*, 36:4213–4224, 2023.

Simon Batzner, Albert Musaelian, Lixin Sun, Mario Geiger, Jonathan P Mailoa, Mordechai Kornbluth, Nicola Molinari, Tess E Smidt, and Boris Kozinsky. E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nature communications*, 13(1):2453, 2022.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

Alberto Bietti and Julien Mairal. Group invariance, stability to deformations, and complexity of deep convolutional representations. *Journal of Machine Learning Research*, 20(25):1–49, 2019.

Deyu Bo, Chuan Shi, Lele Wang, and Renjie Liao. Specformer: Spectral graph neural networks meet transformers. In *The Eleventh International Conference on Learning Representations*, 2023.

Ron Bracewell and Peter B Kahn. The fourier transform and its applications. *American Journal of Physics*, 34(8):712–712, 1966.

Johannes Brandstetter, Rob Hesselink, Elise van der Pol, Erik J Bekkers, and Max Welling. Geometric and physical quantities improve E(3) equivariant message passing. *arXiv preprint arXiv:2110.02905*, 2021.

Johannes Brandstetter, Max Welling, and Daniel E Worrall. Lie point symmetry data augmentation for neural PDE solvers. In *International Conference on Machine Learning*, pp. 2241–2256. PMLR, 2022.

Xavier Bresson and Thomas Laurent. Residual gated graph ConvNets. *arXiv preprint arXiv:1711.07553*, 2017.

M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

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

Dmitri Burago, Iu. D. Burago, and S. B. Ivanov. *A course in metric geometry / Dmitri Burago, Yuri Burago, Sergei Ivanov.* Graduate studies in mathematics vol. 33. American Mathematical Society, Providence, R.I, 2001. ISBN 0821821296.

Dexiong Chen, Leslie O'Bray, and Karsten Borgwardt. Structure-aware transformer for graph representation learning. In *International conference on machine learning*, pp. 3469–3489. PMLR, 2022.

Shuxiao Chen, Edgar Dobriban, and Jane H Lee. A group-theoretic framework for data augmentation. *Journal of Machine Learning Research*, 21(245):1–71, 2020.

Fan RK Chung. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997.

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

Taco Cohen, Maurice Weiler, Berkay Kicanaoglu, and Max Welling. Gauge equivariant convolutional networks and the icosahedral CNN. In *International conference on Machine learning*, pp. 1321–1330. PMLR, 2019.

Taco S Cohen and Max Welling. Steerable CNNs. *arXiv preprint arXiv:1612.08498*, 2016b.

Lynn A Cooper and Roger N Shepard. Chronometric studies of the rotation of mental images. In *Visual information processing*, pp. 75–176. Elsevier, 1973.

Matthieu Cordonnier, Nicolas Keriven, Nicolas Tremblay, and Samuel Vaiter. Convergence of message-passing graph neural networks with generic aggregation on large random graphs. *Journal of Machine Learning Research*, 25(406):1–49, 2024.

Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Veličković. Principal neighbourhood aggregation for graph nets. *Advances in neural information processing systems*, 33: 13260–13271, 2020.

George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

Thomas Dagès, Michael Lindenbaum, and Alfred M Bruckstein. Metric convolutions: A unifying theory to adaptive convolutions. *arXiv preprint arXiv:2406.05400*, 2024.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016a.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NeurIPS*. Curran Associates Inc., 2016b. ISBN 9781510838819.

Congyue Deng, Or Litany, Yueqi Duan, Adrien Poulenard, Andrea Tagliasacchi, and Leonidas J Guibas. Vector neurons: A general framework for SO(3)-equivariant networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12200–12209, 2021.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Weitao Du, He Zhang, Yuanqi Du, Qi Meng, Wei Chen, Nanning Zheng, Bin Shao, and Tie-Yan Liu. SE(3) equivariant graph neural networks with complete local frames. In *International Conference on Machine Learning*, pp. 5583–5608. PMLR, 2022.

Alexandre Duval, Simon V Mathis, Chaitanya K Joshi, Victor Schmidt, Santiago Miret, Fragkiskos D Malliaros, Taco Cohen, Pietro Lio, Yoshua Bengio, and Michael Bronstein. A hitchhiker's guide to geometric gnns for 3D atomic systems. *arXiv preprint arXiv:2312.07511*, 2023a.

Alexandre Agm Duval, Victor Schmidt, Alex Hernández-Garcıa, Santiago Miret, Fragkiskos D Malliaros, Yoshua Bengio, and David Rolnick. Faenet: Frame averaging equivariant gnn for materials modeling. In *International Conference on Machine Learning*, pp. 9013–9033. PMLR, 2023b.

Vijay Prakash Dwivedi, Chaitanya K Joshi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *Journal of Machine Learning Research*, 24 (43):1–48, 2023.

Nadav Dym, Hannah Lawrence, and Jonathan W Siegel. Equivariant frames and the impossibility of continuous canonicalization. In *International Conference on Machine Learning*, pp. 12228–12267. PMLR, 2024.

Carlos Esteves, Christine Allen-Blanchette, Ameesh Makadia, and Kostas Daniilidis. Learning SO(3) equivariant representations with spherical cnns. In *Proceedings of the european conference on computer vision (ECCV)*, pp. 52–68, 2018.

William T Freeman, Edward H Adelson, et al. The design and use of steerable filters. *IEEE Transactions on Pattern analysis and machine intelligence*, 13(9):891–906, 1991.

Fabian Fuchs, Daniel Worrall, Volker Fischer, and Max Welling. SE(3)-transformers: 3D roto-translation equivariant attention networks. *Advances in neural information processing systems*, 33: 1970–1981, 2020.

Mikel Galar, Alberto Fernández, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera. An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. *Pattern Recognition*, 44(8):1761–1776, 2011.

Mario Geiger and Tess Smidt. e3nn: Euclidean neural networks. *arXiv preprint arXiv:2207.09453*, 2022.

Simon Markus Geisler, Arthur Kosmala, Daniel Herbst, and Stephan Günnemann. Spatio-spectral graph neural networks. *Advances in Neural Information Processing Systems*, 37:49022–49080, 2024.

Jan E Gerken, Jimmy Aronsson, Oscar Carlsson, Hampus Linander, Fredrik Ohlsson, Christoffer Petersson, and Daniel Persson. Geometric deep learning and equivariant neural networks. *Artificial Intelligence Review*, 56(12):14605–14662, 2023.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pp. 1263–1272. Pmlr, 2017.

Pablo Gomez, Jennifer Shutter, and Jeffrey N Rouder. Memory for objects in canonical and non-canonical viewpoints. *Psychonomic bulletin & review*, 15(5):940–944, 2008.

John C Gower. Generalized Procrustes analysis. *Psychometrika*, 40(1):33–51, 1975.

Markus Graf. Coordinate transformations in object recognition. *Psychological bulletin*, 132(6):920, 2006.

Jiaqi Han, Yu Rong, Tingyang Xu, and Wenbing Huang. Geometrically equivariant graph neural networks: A survey. *arXiv preprint arXiv:2202.07230*, 2022.

Irina M Harris, Justin A Harris, and Diana Caine. Object orientation agnosia: A failure to find the axis? *Journal of Cognitive Neuroscience*, 13(6):800–812, 2001.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Snir Hordan, Maya Bechler-Speicher, Gur Lifshitz, and Nadav Dym. Spectral graph neural networks are incomplete on graphs with a simple spectrum. *arXiv preprint arXiv:2506.05530*, 2025.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. *Advances in neural information processing systems*, 28, 2015.

Ian T Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.

Sékou-Oumar Kaba and Siamak Ravanbakhsh. Symmetry breaking and equivariant neural networks. *arXiv preprint arXiv:2312.09016*, 2023.

Sékou-Oumar Kaba, Arnab Kumar Mondal, Yan Zhang, Yoshua Bengio, and Siamak Ravanbakhsh. Equivariance with learned canonicalization functions. In *International Conference on Machine Learning*, pp. 15546–15566. PMLR, 2023.

Michael Kazhdan, Thomas Funkhouser, and Szymon Rusinkiewicz. Rotation invariant spherical harmonic representation of 3D shape descriptors. In *Symposium on geometry processing*, volume 6, pp. 156–164, 2003.

Nicolas Keriven and Gabriel Peyré. Universal invariant and equivariant graph neural networks. *Advances in neural information processing systems*, 32, 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

TN Kipf. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4015–4026, 2023.

Miltiadis Kofinas, Boris Knyazev, Yan Zhang, Yunlu Chen, Gertjan J. Burghouts, Efstratios Gavves, Cees G. M. Snoek, and David W. Zhang. Graph neural networks for learning equivariant representations of neural networks. In *The Twelfth International Conference on Learning Representations*, 2024.

Risi Kondor and Shubhendu Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. In *International conference on machine learning*, pp. 2747–2755. PMLR, 2018.

Talia Konkle and Aude Oliva. Canonical visual size for real-world objects. *Journal of Experimental Psychology: human perception and performance*, 37(1):23, 2011.

Devin Kreuzer, Dominique Beaini, Will Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking graph transformers with spectral attention. *Advances in Neural Information Processing Systems*, 34:21618–21629, 2021.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Hannah Lawrence, Vasco Portilheiro, Yan Zhang, and Sékou-Oumar Kaba. Improving equivariant networks with probabilistic symmetry breaking. *arXiv preprint arXiv:2503.21985*, 2025.

Yann LeCun and Yoshua Bengio. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 1998.

Moshe Leshno, Vladimir Ya. Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6(6):861–867, 1993.

Ron Levie, Federico Monti, Xavier Bresson, and Michael M Bronstein. Cayleynets: Graph convolutional neural networks with complex rational spectral filters. *IEEE Transactions on Signal Processing*, 67(1):97–109, 2018.

Heng Li, Zhaopeng Cui, Shuaicheng Liu, and Ping Tan. RAGO: Recurrent graph optimizer for multiple rotation averaging. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15787–15796, 2022.

Yi-Lun Liao and Tess Smidt. Equiformer: Equivariant graph attention transformer for 3D atomistic graphs. In *The Eleventh International Conference on Learning Representations*, 2023.

Derek Lim, Joshua Robinson, Lingxiao Zhao, Tess Smidt, Suvrit Sra, Haggai Maron, and Stefanie Jegelka. Sign and basis invariant networks for spectral graph representation learning. *arXiv preprint arXiv:2202.13013*, 2022.

Derek Lim, Joshua Robinson, Stefanie Jegelka, and Haggai Maron. Expressive sign equivariant networks for spectral geometric learning. *Advances in Neural Information Processing Systems*, 36: 16426–16455, 2023.

Ya-Wei Eileen Lin, Ronen Talmon, and Ron Levie. Equivariant machine learning on graphs with nonlinear spectral filters. *Advances in Neural Information Processing Systems*, 37:128182–128226, 2024a.

Yuchao Lin, Jacob Helwig, Shurui Gui, and Shuiwang Ji. Equivariance via minimal frame averaging for more symmetries and efficiency. *arXiv preprint arXiv:2406.07598*, 2024b.

Peter Lippmann, Gerrit Gerhartz, Roman Remme, and Fred A Hamprecht. Beyond canonicalization: How tensorial messages improve equivariant message passing. *arXiv preprint arXiv:2405.15389*, 2024.

Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8895–8904, 2019.

David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

Shitong Luo, Jiahan Li, Jiaqi Guan, Yufeng Su, Chaoran Cheng, Jian Peng, and Jianzhu Ma. Equivariant point cloud analysis via learning orientations for message passing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18932–18941, 2022.

George Ma, Yifei Wang, and Yisen Wang. Laplacian canonization: A minimalist approach to sign and basis invariant spectral embedding. *Advances in Neural Information Processing Systems*, 36: 11296–11337, 2023.

George Ma, Yifei Wang, Derek Lim, Stefanie Jegelka, and Yisen Wang. A canonicalization perspective on invariant and equivariant learning. *Advances in Neural Information Processing Systems*, 37: 60936–60979, 2024.

Yao Ma, Suhang Wang, Charu C Aggarwal, and Jiliang Tang. Graph convolutional networks with eigenpooling. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 723–731, 2019.

Stephane G Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE transactions on pattern analysis and machine intelligence*, 11(7):674–693, 2002.

Haggai Maron, Heli Ben-Hamu, Nadav Shamir, and Yaron Lipman. Invariant and equivariant graph networks. In *International Conference on Learning Representations*, 2019a.

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

Jonathan Masci, Davide Boscaini, Michael Bronstein, and Pierre Vandergheynst. Geodesic convolutional neural networks on Riemannian manifolds. In *Proceedings of the IEEE international conference on computer vision workshops*, pp. 37–45, 2015.

Sohir Maskey, Ali Parviz, Maximilian Thiessen, Hannes Stärk, Ylli Sadikaj, and Haggai Maron. Generalized Laplacian positional encoding for graph representation learning. *arXiv preprint arXiv:2210.15956*, 2022.

Francesco Mezzadri. How to generate random matrices from the classical compact groups. *arXiv preprint math-ph/0609050*, 2006.

Arnab Kumar Mondal, Siba Smarak Panigrahi, Oumar Kaba, Sai Rajeswar Mudumba, and Siamak Ravanbakhsh. Equivariant adaptation of large pretrained models. *Advances in Neural Information Processing Systems*, 36:50293–50309, 2023.

Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein. Geometric deep learning on graphs and manifolds using mixture model CNNs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5115–5124, 2017.

Christopher Morris, Nils M Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. TUDataset: A collection of benchmark datasets for learning with graphs. *arXiv preprint arXiv:2007.08663*, 2020.

Richard S Palais and Chuu-Lian Terng. A general theory of canonical forms. *Transactions of the American Mathematical Society*, 300(2):771–789, 1987.

Stephen E Palmer. Cannonical perspective and the perception of objects. *Attention and performance*, 9:135–151, 1981.

Jung Yeon Park, Ondrej Biza, Linfeng Zhao, Jan Willem van de Meent, and Robin Walters. Learning symmetric embeddings for equivariant world models. *arXiv preprint arXiv:2204.11371*, 2022.

Saro Passaro and C Lawrence Zitnick. Reducing SO(3) convolutions to SO(2) for efficient equivariant gnns. In *International conference on machine learning*, pp. 27420–27438. PMLR, 2023.

Stefanos Pertigkiozoglou, Evangelos Chatzipantazis, Shubhendu Trivedi, and Kostas Daniilidis. Improving equivariant model training via constraint relaxation. *Advances in Neural Information Processing Systems*, 37:83497–83520, 2024.

Omri Puny, Heli Ben-Hamu, and Yaron Lipman. Global attention improves graph networks generalization. *arXiv preprint arXiv:2006.07846*, 2020.

Omri Puny, Matan Atzmon, Heli Ben-Hamu, Ishan Misra, Aditya Grover, Edward J Smith, and Yaron Lipman. Frame averaging for invariant and equivariant network design. *arXiv preprint arXiv:2110.03336*, 2021.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017a.

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017b.

Guocheng Qian, Hasan Hammoud, Guohao Li, Ali Thabet, and Bernard Ghanem. ASSANet: An anisotropic separable set abstraction for efficient point cloud representation learning. *Advances in Neural Information Processing Systems*, 34:28119–28130, 2021.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Ladislav Rampášek, Michael Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. *Advances in Neural Information Processing Systems*, 35:14501–14515, 2022.

Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *Journal of machine learning research*, 5(Jan):101–141, 2004.

Evan F Risko and Sam J Gilbert. Cognitive offloading. *Trends in cognitive sciences*, 20(9):676–688, 2016.

Olinde Rodrigues. Des lois géométriques qui régissent les déplacements d'un système solide dans l'espace, et de la variation des coordonnées provenant de ces déplacements considérés indépendamment des causes qui peuvent les produire. *Journal de mathématiques pures et appliquées*, 5: 380–440, 1840.

Kusha Sareen, Daniel Levy, Arnab Kumar Mondal, Sékou-Oumar Kaba, Tara Akhound-Sadegh, and Siamak Ravanbakhsh. Symmetry-aware generative modeling through learned canonicalization. *arXiv preprint arXiv:2501.07773*, 2025.

Vıctor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) equivariant graph neural networks. In *International conference on machine learning*, pp. 9323–9332. PMLR, 2021.

Johann Schmidt and Sebastian Stober. Tilt your head: Activating the hidden spatial-invariance of classifiers. In *Forty-first International Conference on Machine Learning*, 2024.

Johann Schmidt and Sebastian Stober. Robust canonicalization through bootstrapped data realignment. *arXiv preprint arXiv:2510.08178*, 2025.

Kristof T Schütt, Farhad Arbabzadah, Stefan Chmiela, Klaus R Müller, and Alexandre Tkatchenko. Quantum-chemical insights from deep tensor neural networks. *Nature communications*, 8(1): 13890, 2017.

Claude E Shannon. Communication in the presence of noise. *Proceedings of the IRE*, 37(1):10–21, 2006.

Roger N Shepard and Jacqueline Metzler. Mental rotation of three-dimensional objects. *Science*, 171 (3972):701–703, 1971.

Ken Shoemake. Uniform random rotations. In *Graphics Gems III (IBM Version)*, pp. 124–132. Elsevier, 1992.

Zakhar Shumaylov, Peter Zaika, James Rowbottom, Ferdia Sherry, Melanie Weber, and Carola-Bibiane Schönlieb. Lie algebra canonicalization: Equivariant neural operators under arbitrary lie groups. In *The Thirteenth International Conference on Learning Representations*, 2025.

Utkarsh Singhal, Ryan Feng, Stella X. Yu, and Atul Prakash. Test-time canonicalization by foundation models for robust perception. In *Forty-second International Conference on Machine Learning*, 2025.

Daniel Spielman. Spectral graph theory. *Combinatorial scientific computing*, 18(18), 2012.

Andrea Tacchetti, Leyla Isik, and Tomaso A Poggio. Invariant recognition shapes neural representations of visual input. *Annual review of vision science*, 4(1):403–422, 2018.

Behrooz Tahmasebi and Stefanie Jegelka. Generalization bounds for canonicalization: A comparative study with group averaging. In *The Thirteenth International Conference on Learning Representations*, 2025a.

Behrooz Tahmasebi and Stefanie Jegelka. Regularity in canonicalized models: A theoretical perspective. In *The 28th International Conference on Artificial Intelligence and Statistics*, 2025b.

Michael J Tarr and Steven Pinker. Mental rotation and orientation-dependence in shape recognition. *Cognitive psychology*, 21(2):233–282, 1989.

Erik Henning Thiede, Truong Son Hy, and Risi Kondor. The general theory of permutation equivarant neural networks and higher order graph variational encoders. *arXiv preprint arXiv:2004.03990*, 2020.

Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation-and translation-equivariant neural networks for 3D point clouds. *arXiv preprint arXiv:1802.08219*, 2018.

Lloyd N Trefethen and David Bau. *Numerical linear algebra*. SIAM, 2022.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Clement Vignac, Andreas Loukas, and Pascal Frossard. Building powerful and equivariant graph neural networks with structural message-passing. *Advances in neural information processing systems*, 33:14143–14155, 2020.

Soledad Villar, David W Hogg, Kate Storey-Fisher, Weichi Yao, and Ben Blum-Smith. Scalars are universal: Equivariant machine learning, structured like classical physics. *Advances in neural information processing systems*, 34:28848–28863, 2021.

Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

Edward Wagstaff, Fabian B Fuchs, Martin Engelcke, Michael A Osborne, and Ingmar Posner. Universal approximation of functions on sets. *Journal of Machine Learning Research*, 23(151): 1–56, 2022.

Rui Wang, Robin Walters, and Rose Yu. Approximately equivariant networks for imperfectly symmetric dynamics. In *International Conference on Machine Learning*, pp. 23078–23091. PMLR, 2022.

Rui Wang, Elyssa Hofgard, Han Gao, Robin Walters, and Tess E Smidt. Discovering symmetry breaking in physical systems with relaxed group convolution. *arXiv preprint arXiv:2310.02299*, 2023.

Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph CNN for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5): 1–12, 2019.

Maurice Weiler and Gabriele Cesa. General E(2)-equivariant steerable CNNs. *Advances in neural information processing systems*, 32, 2019.

Maurice Weiler, Mario Geiger, Max Welling, Wouter Boomsma, and Taco S Cohen. 3D steerable CNNs: Learning rotationally equivariant features in volumetric data. *Advances in Neural information processing systems*, 31, 2018a.

Maurice Weiler, Fred A Hamprecht, and Martin Storath. Learning steerable filters for rotation equivariant CNNs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 849–858, 2018b.

Maurice Weiler, Patrick Forré, Erik Verlinde, and Max Welling. Coordinate independent convolutional networks–isometry and gauge equivariant convolutions on Riemannian manifolds. *arXiv preprint arXiv:2106.06020*, 2021.

Ruben Wiersma, Ahmad Nasikun, Elmar Eisemann, and Klaus Hildebrandt. DeltaConv: anisotropic operators for geometric deep learning on point clouds. *ACM Transactions on Graphics (TOG)*, 41 (4):1–10, 2022.

Daniel E Worrall, Stephan J Garbin, Daniyar Turmukhambetov, and Gabriel J Brostow. Harmonic networks: Deep translation and rotation equivariance. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5028–5037, 2017.

Zhanghao Wu, Paras Jain, Matthew Wright, Azalia Mirhoseini, Joseph E Gonzalez, and Ion Stoica. Representing long-range context for graph neural networks with global attention. *Advances in neural information processing systems*, 34:13266–13279, 2021.

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

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (ToG)*, 35(6):1–12, 2016.

Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. *Advances in neural information processing systems*, 31, 2018.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.

Yan Zhang, Jonathon Hare, and Adam Prügel-Bennett. Learning representations of sets through optimized permutations. *arXiv preprint arXiv:1812.03928*, 2018.

Yan Zhang, Jonathon Hare, and Adam Prügel-Bennett. FSPool: Learning set representations with featurewise sort pooling. *arXiv preprint arXiv:1906.02795*, 2019a.

Zhen Zhang, Jiajun Bu, Martin Ester, Jianfeng Zhang, Chengwei Yao, Zhi Yu, and Can Wang. Hierarchical graph pooling with structure learning. *arXiv preprint arXiv:1911.05954*, 2019b.

Appendix

A Related Work

We provide an extended discussion of related work for equivariant machine learning.

A.1 Canonicalization

Canonicalization (Babai & Luks, 1983; Palais & Terng, 1987; Ma et al., 2023; 2024) is a classical strategy for handling symmetry in data, especially for tasks where invariance or equivariance to group actions is desirable (Gerken et al., 2023). It preprocesses each input by mapping it to a standard form prior to downstream learning and inference, so that all symmetry-equivalent inputs are treated identically by the subsequent model. There are two common ways for canonicalization: fixed or learned approaches.

- **Fixed Canonicalization.** Fixed canonicalization uses deterministic and often analytic procedures to assign a unique representative to each symmetry orbit. For example, principal component analysis (PCA) (Jolliffe & Cadima, 2016) alignment canonically orients an object (e.g., a point cloud or molecule) by rotating it so its principal components align with the coordinate axes (Kazhdan et al., 2003). Procrustes analysis (Gower, 1975) canonically orients sets of points by finding the optimal rotation, translation, and scale that minimizes squared point-to-point distances to a reference. For sets and graphs, canonicalization can be achieved by reordering nodes, atoms, or features so that isomorphic inputs share a single labeling. In spectral methods and spectral graph neural networks where eigenvectors are fundamental (Kipf, 2016; Defferrard et al., 2016a; Von Luxburg, 2007; Belkin & Niyogi, 2003; Dwivedi et al., 2023; Maskey et al., 2022), canonicalization of spectral decomposition (Lim et al., 2022; 2023; Ma et al., 2023; 2024) addresses eigenbasis ambiguity (Chung, 1997; Spielman, 2012) by processing each eigenspace independently and selecting representative eigenvectors or directions by applying orthogonal or axis-based projections, typically as a graph preprocessing step. An alternative approach is eigenbasis canonicalization via the input signal, where the signal itself is used to define a canonical spectral representation, making the spectral transformation independent of the arbitrary choice of eigenvectors (Lin et al., 2024a; Geisler et al., 2024).
- **Learned Canonicalization.** Learned canonicalization (Zhang et al., 2018; Kaba et al., 2023; Luo et al., 2022) seeks to overcome the rigidity and inflexibility of fixed rules with a trainable mapping that selects a representative for each symmetry orbit. The canonicalizer is parameterized (typically as a neural network and trained to produce canonical forms. For example, Kaba et al. (2023) developed a neural network that learns the canonicalization transformation, which enables plug-and-play equivariance, e.g., orthogonalizing learned features via the Gram-Schmidt process (Trefethen & Bau, 2022). Their results show that the learned canonicalizers outperform fixed canonicalizers.

However, Dym et al. (2024) pointed out that regardless of whether the canonicalizations are learned or not, a continuous canonicalization does not exist for many common groups (e.g. S$_n$, SO(d), O(d) on point clouds $n \geq d$). Therefore, while learned canonicalization improves empirical performance, it remains generally discontinuous and can induce instability, hinder generalization, limit model reliability on boundary cases, or out-of-distribution data. In contrast, our adaptive canonicalization framework learns the optimal transformation for each input by maximizing the predictive confidence of the network, resulting in a continuous and symmetry preserving mapping. We include a detailed comparison of most related canonicalization work with our method below.

A.1.1 Equivariance with Learned Canonicalization Functions

Recently, Kaba et al. (2023) introduced energy-based canonicalization. The central idea is to learn an energy function over samples and group elements, and define the canonicalization as minimizing this energy with respect to the group, given a fixed datapoint. Specifically, their energy minimization is related to our prior maximization adaptive canonicalization method. However, there are several important conceptual and technical differences between their approach and our adaptive canonicalization.

First, their energy s is not the task neural network like in our analysis, but rather some other trainable neural network. Similarly to our approach, training s end-to-end with the task neural network can be seen as canonicalization that depends on the task network, if one considers the full end-to-end architecture consisting both of the energy minimization and the task network. However, the approach in Kaba et al. (2023) does not give continuity guarantees as opposed to our approach. Second, in their work, they consider symmetries based on group actions, while we consider a more general setting of canonicalization transformations (or augmentations) that need not be based on groups. This makes our approach much more applicable across different domains. Moreover, in their framework, there is one canonical form for each datapoint and network, while in our approach, each output channel of the network defines a different canonical form of the datapoint. This allows our approach to preserve continuity. Notably, Kaba et al. (2023) do not attempt to study the continuity of the end-to-end predictor.

Another difference in Kaba et al. (2023) is that when training is initialized, the canonicalizing energy s is random. This leads each datapoint to be randomly transformed, so the task neural network initially has to perform well at all orientations of the data. This can lead the task network to ultimately learn an “average behavior,” not specializing in any special orientation but rather performing reasonably well on all orientations of the data. In other words, the limited set of trainable parameters has to simultaneously specialize in many orientations, which reduces the network’s expressive power. In contrast, in our prior maximization approach, from the beginning of training, the network only pays a price for not performing well on the single best orientation per datapoint (on which the network performs the best). This encourages the network to specialize on one canonical orientation per datapoint, and not learn an average behavior. Hence, in our approach, all trainable parameters of the task network can focus on performing well only on the sole canonical orientation of each datapoint.

A.1.2 Canonicalization and Data Re-Alignments

As discussed in App. A.1.1, in the energy-based canonicalization framework of (Kaba et al., 2023), the canonicalizing energy s is random at initialization. As a result, it leads each datapoint to be randomly transformed and the task neural network initially has to perform well at all orientations. This can lead the task network to ultimately learn an “average behavior,” not specializing in any special orientation but rather performing reasonably well on all orientations of the data. To address this effect, Mondal et al. (2023) biases the canonical transformation of each datapoint to be the identity, assuming that the datapoints in the training set already have a small orientation variance. On top of that, Schmidt & Stober (2025) iteratively reduces the orientation variance of the training set by iteratively reorienting datapoints that lead to a large loss. We note that these approaches are rather different from our prior maximization method, and they do not try to address the continuity problem in canonicalization.

A.1.3 Weighted Canonicalization

The energy-based canonicalization (Kaba et al., 2023) was further explored and extended in (Shumaylov et al., 2025) on symmetries defined by general Lie group actions. Similarly to our work, Shumaylov et al. (2025) also discusses continuity preservation, but their approach is different from ours. In the work of Shumaylov et al. (2025), they define the notion of weighted canonicalization, which is a similar concept to the weighted frame introduced by Dym et al. (2024). Here, to each datapoint there is an assigned probability measure over the orbit of the datapoint. Namely, the distribution is over the space of data instead of over the group like in the weighted frames of Dym et al. (2024). With respect to energy minimization, this approach is not very different from Kaba et al. (2023). The main difference is in the minimization algorithm, which minimizes over the Lie algebra instead of the Lie group. It is important to note that their work does not train the energy end-to-end with the neural network. Hence, in their work, the canonicalization depends on the whole training set, but not on the task network, which is quite different from our approach. Moreover, in their setup, one has to learn an approximation of the data distribution, which is invariant to the group action. This is a highly nontrivial approach to implement. In contrast, our prior maximization is simple and direct.

A.1.4 Test-Time Canonicalization

Another recent extension of canonicalization, (Singhal et al., 2025) , explores a set of transformations at test time and uses the scoring functions of large pre-trained foundation models like CLIP (Radford et al., 2021) or SAM (Kirillov et al., 2023) to select the most "canonical" representation upon which downstream inference is performed. Note that the work in Singhal et al. (2025) does not involve network retraining, uses the foundation models as is, and performs canonicalization entirely at inference by optimizing over transformations. While it achieves strong empirical performance, its canonicalization mapping is not guaranteed to be continuous, and in fact, continuity is not discussed. Therefore, small input changes may cause abrupt switches in the selected canonical view.

A related line of work is inverse transformation search (Schmidt & Stober, 2024), which also performs test-time optimization over transformations to exploit invariances. Their method focuses on the standard action of the special linear group on images (rotations, scalings, and shear transformations), i.e., purely group-based symmetries, and similar to Singhal et al. (2025), does not address continuity. In addition, their method does not train the model simultaneously with the canonicalization. While their energy-induced confidence is similar to our prior-maximization formulation in the classification setting, it does not lead to a continuity guarantee. In contrast, our work rigorously develops sufficient conditions on the canonicalizer for the canonicalized network to be continuous and have a universal approximation property. Specifically, our one-vs.-all setting is a different type of energy that does lead to continuous end-to-end classifiers.

A.2 Frame Averaging

Frame averaging (Puny et al., 2021) achieves equivariance to group symmetries by averaging a network's output over a set of group transformations (known as a "frame"). It is built on the classical group averaging operator, which guarantees symmetries by summing a function over all group elements. Frame averaging has two main advantages: 1) it allows adaptation of standard non-equivalent network architectures to handle symmetry, similar to canonicalization methods, and 2) it avoids computational intractability of full group averaging, especially for large or continuous groups. Recent work (Lin et al., 2024b) proposes minimal frame averaging that attains strong symmetry coverage with small frames. Domain-specific frame averaging methods (Duval et al., 2023b; Atzmon et al., 2022) show that it can be deployed in material modeling and geometric shape analysis. However, it requires a careful selection of a suitable frame. In addition, frame averaging uses a fixed set of transformations independent of the input or task, potentially leading to sub-optimal or less discriminative feature representations. In contrast, our adaptive canonicalization learns the optimal transformation for each input in a data- and network-dependent way, yielding symmetry preserving continuous functions that can improve representation quality and empirical task performance.

A related work that addresses continuity is the weighted frame averaging proposed by Dym et al. (2024). They first prove that in many well-known cases, continuous canonicalization is impossible. This does not contradict our work, as in Dym et al. (2024) the canonicalization is a function solely of the datapoint, and not the task network. Then, they define a variant of frame averaging, called weighted frame averaging, in which to each datapoint there is an associated probability distribution over the group, and the frame averaging is performed with respect to this measure. This construction yields continuity guarantees. However, their focus is fundamentally different from ours: they study frame averaging while we focus on canonicalization. Moreover, the weighted frame averaging is a function only of the data, not the neural network, as opposed to our method. We next compare our framework to weighted frame averaging in more detail.

- In our method, data need not come from a vector space of a fixed dimension (see e.g., the application for graphs). In contrast, weighted frame averaging requires working with data that comes from a vector space of a fixed dimension.
- In our work, symmetries need not be based on group representations. Our notion of symmetry is called a transformation family, and our "symmetries" are not even required to be invertible or based on a group action. See, for example, the image truncation transformation in App. E.5. On the other hand, the symmetries in weighted frame averaging are required to be representations of compact groups.
- In the framework of weighted frame averaging, the requirement that the weighted frame is robust is sufficient for continuity of the canonicalized function. However, this requirement is quite strong, and constructing robust frames could be challenging. In contrast, we achieve continuity of the canonicalized function even though the maximizer in prior maximization need not be continuous in any sense (as a function of the datapoint). The maximizer need not even be uniquely defined. Hence, practitioners can use our method out of the box on new

domains with new symmetries without having to prove any nontrivial mathematical results. In weighted frame averaging, employing the method in a new setting typically requires carefully constructing a problem-specific weighted frame and proving that it is robust. It often involves nontrivial mathematical work and there is no general recipe for doing so. In our case, once an architecture and a symmetry setting are chosen, prior-maximization adaptive canonicalization is straightforward to implement and does not require additional sophisticated proofs from the practitioner. The only assumption that needs to be checked is continuity of the chosen transformations, which is usually easy to verify.
- Robust weighted frames often require averaging the predicting network over many transformations of the input. In practice, our prior maximization approach works with fewer random argmax candidates. For comparison, frame averaging for rotations of n points in a 3D point cloud requires an order of n^2 transformations (where $n = 1024$ in the ModelNet40 dataset for example), while, in practice, prior maximization performs well with a total of 50 random transformations.

These differences imply that our framework is more flexible in terms of the data types and symmetry structures it can accommodate, while also imposing a lighter mathematical burden on practitioners who wish to apply it. This gives much more freedom for future practitioners, and may lead to a wider adaptation of the method.

We note that the earlier work (Basu et al., 2023) proposed a similar idea to weighted frame averaging, but did not study continuity preservation.

A.3 Equivariant Architectures

Equivariant architectures are a class of models explicitly designed to respect symmetry groups acting on the data. Formally, a network is equivariant to a group of transformations if, when the input is transformed by a symmetry group action, the output transforms via the same group action. That is, for a group G and a function f, equivariance guarantees $f(g \cdot x) = g \cdot f(x)$ for all $g \in G$ and input x. It has been developed across images (Cohen & Welling, 2016a;b; Kondor & Trivedi, 2018; Worrall et al., 2017; Weiler et al., 2018b), graphs (Bronstein et al., 2017; Zaheer et al., 2017; Gilmer et al., 2017; Maron et al., 2019a; Kofinas et al., 2024; Thiede et al., 2020; Vignac et al., 2020; Keriven & Peyré, 2019), molecules (Thomas et al., 2018; Brandstetter et al., 2021; Anderson et al., 2019; Fuchs et al., 2020; Satorras et al., 2021; Duval et al., 2023a; Schütt et al., 2017; Liao & Smidt, 2023; Hordan et al., 2025; Du et al., 2022; Passaro & Zitnick, 2023), and manifolds (Masci et al., 2015; Monti et al., 2017; Cohen et al., 2019; Weiler et al., 2021). Notably, Maron et al. (2019b) studies the universal approximation property for equivariant architectures. Notably, equivariant networks often demand group-specific designs that rely on group theory, representation theory, and tensor algebra. This can reduce flexibility and raise compute and memory costs (Pertigkiozoglou et al., 2024; Wang et al., 2023; Liao & Smidt, 2023). Making nonlinearities, pooling, and attention strictly equivariant further constrains layer choices and can increase parameter count and runtime. Moreover, imposing symmetry throughout the stack may limit the expressivity when data only approximately respect the assumed symmetry or contain symmetry-breaking noise (Wang et al., 2022; Lawrence et al., 2025). In contrast, our approach handles symmetry by learning an input- and task-dependent canonical form through prior maximization and apply a standard backbone. By construction, this mapping is continuous and symmetry preserving. Our adaptive canonicalization removes heavy group-specific layers, reduces per-layer equivariant cost, and keeps ordinary nonlinearities and pooling.

B Universal Approximation Theorems

B.1 Universal Approximation of Euclidean Fucntions

Here, we cite a classical result stating that multilayer perceptrons (MLPs) are universal approximators of functions over compact sets in Euclidean spaces.

Theorem 14 (Universal Approximation Theorem (Leshno et al., 1993))**.** *Let $\sigma : \mathbb{R} \to \mathbb{R}$ be a continuous, non-polynomial function. Then, for every $M, L \in \mathbb{N}$, compact $\mathcal{K} \subseteq \mathbb{R}^M$, continuous function $f : \mathcal{K} \to \mathbb{R}^L$, and $\varepsilon > 0$, there exist $D \in \mathbb{N}$, $\boldsymbol{W}_1 \in \mathbb{R}^{D\times M}$, $\boldsymbol{b}_1 \in \mathbb{R}^D$ and $\boldsymbol{W}_2 \in \mathbb{R}^{L\times D}$ s.t.*

$$\sup_{\boldsymbol{x}\in\mathcal{K}} |f(\boldsymbol{x}) - \boldsymbol{W}_2\sigma(\boldsymbol{W}_1\boldsymbol{x} + \boldsymbol{b}_1)| \leq \varepsilon. \tag{4}$$

B.2 Universal Approximation of Multi-Sets Functions

Multisets are sets where repetitions of elements are allowed. Formally, a multiset of N elements in $\mathbb{R}^J$ can be defined as a set of pairs (x, i) where x denotes a point and i the number of times the point x appears in the multiset.

Standard universal approximation analysis of multi-set functions goes along the following lines. First, we represent multisets as 2D arrays $\mathbf{X} \in \mathbb{R}^{N \times J}$, where each row $\mathbf{X}_{n,:} \in \mathbb{R}^J$ represents one point in the multiset. Note that the same multi-set can be represented by many arrays. In fact, two arrays $\mathbf{X}$ and $\mathbf{X}'$ represent the same multi-set if and only if one is a permutation of the other.

To formulate this property, let $\mathcal{S}_N$ be the symmetric group of N elements, i.e., the group of permutations of N elements. Given $s \in \mathcal{S}$ and $\mathbf{X} \in \mathbb{R}^{N \times J}$, let $\rho(s)\mathbf{X}$ denote the permutation of $\mathbf{X}$ via s. By convention, permutations change the order of the N rows of $\mathbf{X}$, and keeps each row intact. Now, $\mathbf{X}$ and $\mathbf{X}'$ represent the same multi-set if and only if there is a permutation $s \in \mathcal{S}_N$ such that $\mathbf{X} = \rho(s)\mathbf{X}'$.

Now, for a function $y : \mathbb{R}^{N \times J} \to \mathbb{R}^D$ to represent a multi-set function, it should be invariant to permutations, i.e., for every $s \in \mathcal{S}_N$ we have $y(\rho(s)\mathbf{X}) = y(\mathbf{X})$. Hence, standard UATs of multi-set functions are formulated based on the following notion of universal approximation.

Definition 15. *Let $\mathcal{K} \subset \mathbb{R}^{N \times J}$ be an invariant compact domain, i.e., $\rho(s)\mathbf{X} \in \mathcal{K}$ for every $\mathbf{X} \in \mathcal{K}$ and $s \in \mathcal{S}_N$. A set of invariant functions $\mathcal{N}(\mathcal{K}, \mathbb{R}^D) \subset C_0(\mathcal{K}, \mathbb{R}^D)$ is called an* invariant universal approximator of $C_0(\mathcal{K}, \mathbb{R}^D)$ equivariant functions *if for every invariant function $y \in C_0(\mathcal{K}, \mathbb{R}^D)$ and $\epsilon > 0$ where is $\theta \in \mathcal{N}(\mathcal{K}, \mathbb{R}^D)$ such that for every $\mathbf{X} \in \mathcal{K}$*

$$|\theta(\mathbf{X}) - y(\mathbf{X})| < \epsilon.$$

For example, DeeptSets are universal approximators of invariant $C_0(\mathcal{K}, \mathbb{R}^D)$ functions (Wagstaff et al., 2022).

In this section, we describe an alternative, but equivalent, approach to model multisets of size N and their universal approximation theorems using the notion of quotient. The motivation is that our main UAT, Theorem 13, is based on the standard symmetryless notion of universal approximation, Definition 3. While it is possible to develop our adaptive canonicalization theory for functions that preserve symmetries, and obtain an analogous theorem to Theorem 13 based on invariant universal approximators, there is no need for such complications. Instead, we can use the standard definition of universal approximation (Definition 3), and directly encode the symmetries in the domain using quotient spaces, as we develop next.

Quotient Spaces. Let $\mathcal{X}$ be a topological space, and $x \sim y$ an equivalence relation between pairs of points. The equivalence class $[x]$ of $x \in \mathcal{X}$ is defined to be the set

$$[x] := \{y \in X \ x \sim y\}.$$

Definition 16 (Quotient topology)**.** *Let $\mathcal{X}$ be a topological space, and $\sim$ an equivalence relation on $\mathcal{X}$. The* quotient set *is defined to be*

$$\mathcal{X}/\sim := \{[x] \mid x \in \mathcal{X}\}.$$

The quotient set is endowed with the quotient topology*. The quotient topology is the finest (largest) topology making the mapping $\nu : x \mapsto [x]$ continuous. In other words, the open sets $B \subset (\mathcal{X}/\sim)$ are those sets such that $\cup_{[x] \in B}[x]$ is open in $\mathcal{X}$.*

The mapping $\nu : \mathcal{X} \to (\mathcal{X}/\sim)$, defined by $\nu(x) = [x]$, is called the *canonical projection*.

Multi-Sets as Equivalence Classes and UATs. Define the equivalence relation: $\mathbf{X} \sim \mathbf{Y}$ if there exists $s \in \mathcal{S}_N$ such that $\mathbf{X} = \rho(s)\mathbf{Y}$. Now, a multi-set of N elements can be defined as $\mathbb{R}^{N \times J}/\sim$. As opposed to the definition of multisets as sets of pairs (x, i), the quotient definition automatically gives a topology to the sets of multisets, namely, the quotient topology. In fact, it can be shown that the quotient topology is induced by the following metric.

Definition 17 (Multi-Set Metric). *Given $[\mathbf{X}], [\mathbf{Y}] \in (\mathbb{R}^{N \times J}/\sim)$, their distance is defined to be*

$$d([\mathbf{X}], [\mathbf{Y}]) := \min_{\mathbf{X}' \in [\mathbf{X}], \mathbf{Y}' \in [\mathbf{Y}]} \|\mathbf{X}' - \mathbf{Y}'\|.$$

It is easy to see that

$$d([\mathbf{X}],[\mathbf{Y}]) := \min_{s\in\mathcal{S}_N} \|\mathbf{X}-\rho(s)\mathbf{Y}\|.$$

One can show that the above distance is indeed a metric, and that the topology induced by this metric is exactly the quotient topology, i.e., d *metrizes* $(\mathbb{R}^{N\times J}/\sim)$.

Theorem 18. *The metric $d([\mathbf{X}],[\mathbf{Y}])$ metrizes the quotient topology $\mathbb{R}^{N\times J}/\sim$.*

A set $\mathcal{K}\subset\mathbb{R}^{N\times J}$ is called invariant if for every $\mathbf{X}\in\mathcal{K}$ and $s\in\mathcal{S}_N$ we have $\rho(s)\mathbf{X}\in\mathcal{K}$. Consider the quotient space

$$\mathcal{K}/\sim=\{[\mathbf{X}]\mid\mathbf{X}\in\mathcal{K}\}\subset\mathbb{R}^{N\times J}/\sim.$$

We now have the following proposition about continuity of symmetric functions.

Proposition 19. *Let $\mathcal{K}\subset\mathbb{R}^{N\times J}$ be an invariant compact domain. For every continuous invariant mapping $y:\mathcal{K}\to\mathbb{R}^D$ there exists a unique continuous mapping $\overline{y}:(\mathcal{K}/\sim)\to\mathbb{R}^D$ such that*

$$y=\overline{y}\circ\nu,$$

where ν is the canonical projection. On the other hand, for every function $z\in C_0(\mathcal{K}/\sim,\mathbb{R}^D)$, we have that $z\circ\nu$ is a continuous invariant function in $C_0(\mathcal{K},\mathbb{R}^D)$.

Let $\mathcal{K}\subset\mathbb{R}^{N\times j}$ be an invariant domain. For a set of continuous invariant functions $\mathcal{N}\subset C_0(\mathcal{K},\mathbb{R}^D)$, we denote

$$(\mathcal{N}/\sim):=\{\overline{y}\mid y\in\mathcal{N}\}\subset C_0(\mathcal{K}/\sim,\mathbb{R}^D).$$

Note that by Proposition 19

$$\big(C_0(\mathcal{K},\mathbb{R}^D)/\sim\big)=C_0\big(\mathcal{K}/\sim,\mathbb{R}^D\big).$$

This immediately leads to a UAT theorem for multi-set continuous functions in which every continuous multi-set function can be approximated by a neural network.

Theorem 20. *Let $\mathcal{K}\subset\mathbb{R}^N$ be an invariant compact domain, and let $\mathcal{N}(\mathcal{K},\mathbb{R}^D)\subset C_0(\mathcal{K},\mathbb{R}^D)$ be an invariant universal approximator of $C_0(\mathcal{K},\mathbb{R}^D)$ equivariant functions. Then $\mathcal{N}(\mathcal{K},\mathbb{R}^D)/\sim$ is a universal approximator $C_0(\mathcal{K}/\sim,\mathbb{R}^D)$.*

Note that in the above UAT the symmetries are directly encoded in the quotient spaces, and, hence, there is no need to encode any symmetry in the spaces of functions. Hence, Theorem 20 is based on the standard symmetryless definition of universal approximation – Definition 3 – rather than the symmetry driven construction of Definition 15. As a result, we can directly use our theory of adaptive canonicalization on multi-set functions. Specifically, we can use Theorem 13, where the space $\mathcal{K}$ in the theorem is taken as $\mathcal{K}/\sim$ in our above analysis.

Now, we immediately obtain that the set of neural networks $\overline{\theta}$ where θ is a DeepSet is a universal approximator of the space of continuous multi-set functions.

C Additional Examples of Continuous Prior Maximization

We first note that when $\mathcal{K}$ is a locally compact metric space, functions in $C_0(\mathcal{K},\mathbb{R}^D)$ must be uniformly continuous.

Unbounded Point Clouds and Rotations. Let $\mathcal{U}=\mathcal{SO}(3)$ be the space of 3D rotations, and $\mathcal{G}=\mathcal{K}=\mathbb{R}^{N\times 3}$ the set of sequences of N points in $\mathbb{R}^3$, i.e. the space of point clouds. Consider the $\mathcal{L}_2$ metric in $\mathcal{G}$. Consider the rotation $g\mapsto\kappa_u(g)$ of the point cloud g by $u\in\mathcal{U}$.

Let $f\in C_0(\mathcal{G},\mathbb{R}^D)$. Next we show that f must be uniformly continuous. Let $\epsilon>0$. By the fact that f vanishes at infinity, there exists a compact domain $\mathcal{K}\subset\mathbb{R}^{N\times 3}$ such that for every $x\notin\mathcal{K}$ we have $|f(x)|<\epsilon/2$. By the fact that $\mathcal{U}\times\mathcal{K}$ is compact and κ continuous, κ is uniformly continuous on $\mathcal{U}\times\mathcal{K}$. Hence, there exists $\delta_\epsilon>0$ such that every $g,g'\in\mathcal{K}$ with $d(g,g')<\delta_\epsilon$ satisfy $d(f(g),f(g'))<\epsilon$. Let κ' be the compact space consisting of all point of distance less or equal to δ_ϵ from $\mathcal{K}$. There exists $0<\delta'_\epsilon<\delta_\epsilon$ such that every $g,g'\in\mathcal{K}'$ with $d(g,g')<\delta'_\epsilon$ satisfy $d(f(g),f(g'))<\epsilon$.

Now, let $g,g'\in\mathcal{G}$ satisfy $d(g,g')<\delta'_\epsilon$. If one of the point g or g' lies outside $\mathcal{K}'$, then both of them lie outside $\mathcal{K}$, so

$$d(f(g),f(g'))\leq\|f(g)\|+\|f(g')\|<\epsilon.$$

Otherwise, both lie in $\mathcal{K}'$, so $d(f(g), f(g')) < \epsilon$. Both cases together mean that f is uniformly continuous.

As a result of uniform continuity, $\{g \mapsto f(\kappa_u(g))\})u \in \mathcal{U}$ is equicontinuous, and this is a setting of continuous prior maximization.

In fact, this analysis shows that whenever $\mathcal{G} = \mathcal{K}$ and κ is continuous in (u, g), then it's corresponding ρ is continuous prior maximization.

Continuous to Discrete Images with Rotations and Other Image Transformations. Consider the "continuous" space of images $\mathcal{G} = \mathcal{L}_2(\mathbb{R}^2)$ and the discrete space $\mathcal{K} = \mathbb{R}^{N\times N}$ of images of $N \times N$ pixels.

Let $\mathcal{U}$ be the unit circle. For $g \in \mathcal{G}$ and $u \in \mathcal{U}$ let $\pi(u)g$ be the rotation of the image g by angle u. To define the discretizing mapping $P : \mathcal{L}_2(\mathbb{R}^2) \to \mathbb{R}^{N\times N}$, consider the partition of $[-1, 1]$ into the N intervals

$$I_n = [-1 + 2n/N, -1 + 2(n + 1)/N), \quad n = 0, \dots N - 1.$$

Consider the closed linear subspace $\mathcal{D}(\mathbb{R}^2) \subset \mathcal{L}_2(\mathbb{R}^2)$ consisting of images that are zero outside $[-1, 1]^2$ and piecewise constant on the squares $\{I_n \times I_m\}_{n,m=0}^{N-1}$. Now, P is the operator that takes $g \in \mathcal{L}_2(\mathbb{R}^2)$ first orthogonally projects it upon $\mathcal{D}(\mathbb{R}^2)$ to get g', and then returns

$$P(g) = \{g'(-1 + 2n/N, -1 + 2m/N)\}_{n,m=0}^{N-1} \in \mathbb{R}^{N\times N}.$$

Define the mapping κ_u as follows: $\kappa_u(g) = P(\pi(u)g)$. By the fact that π_u is an isometry for every $u \in \mathcal{U}$ and P is non-expansive (as an orthogonal projection), $\kappa_u : \mathcal{G} \to \mathbb{R}^{N\times N}$ is Lipschitz 1 for every u. Hence, $\{g \mapsto f(\kappa_u(g))\}_{u\in\mathcal{U}}$ is equicontinuous.

As a result, the corresponding prior minimization is a continuous prior minimization.

This setting can be extended to other image deformation based on diffeomorphisms of the domain $\mathbb{R}^2$, parameterized by compact spaces $\mathcal{U}$, For example, one can take dilations up to some uniformly bounded scale. More generally, one can consider a compact set $\mathcal{U}$ of matrices in $\mathbb{R}^{2\times 2}$, and define $\pi(u)g(\mathbf{x}) = g(\S\mathbf{U})$ for $\mathbf{U} \in \mathcal{U}$, $g \in L_2(\mathbb{R}^2)$ and $\mathbf{x} \in \mathbb{R}^2$.

Discrete to Discrete Images with Rotations and Other Image Transformations. One can replace $\mathcal{G}$ by $\mathcal{D}(\mathbb{R}^2)$ in the above analysis. Since now $\mathcal{D}(\mathbb{R}^2)$ is compact, ρ must be a continuous prior maximization.

The above examples can be naturally extended to additional image transformation, like translations and dilations. Notably, translations and dilations do not form a compact group, but still they satisfy the conditions of our theory, which requires no compactness assumptions.

D Functional Calculus and Spectral Filters

In this section, we recall the theory of plugging self-adjoint operators inside functions.

Spectral Theorem. Let $\mathcal{L}$ be a self-adjoint operator on a finite-dimensional Hilbert space (e.g., $\mathcal{L} \in \mathbb{C}^{N\times N}$ with $\mathcal{L} = \mathcal{L}^*$). There exists a unitary matrix V and a real diagonal matrix $\Lambda = \operatorname{diag}(\lambda_1, \dots, \lambda_N)$ such that $\mathcal{L} = V\Lambda V^*$. The columns v_i of V form an orthonormal eigenbasis with $\mathcal{L}v_i = \lambda_i v_i$.

Functional Calculus. For any function $f : \mathbb{R} \to \mathbb{C}$ defined on the spectrum $\sigma(\mathcal{L}) = \{\lambda_1, \dots, \lambda_N\}$:

$$f(\mathcal{L}) \;:=\; Vf(\Lambda)V^*, \qquad f(\Lambda) = \operatorname{diag}\big(f(\lambda_1), \dots, f(\lambda_N)\big).$$

Equivalently, we can write the spectral projections $P_i := v_i v_i^*$,

$$f(\mathcal{L}) \;=\; \sum_{i=1}^{N} f(\lambda_i)\, P_i.$$

We can plug a self-adjoint matrix into a function by: 1) diagonalize $\mathcal{L}$, 2) apply f to the eigenvalues, and 3) conjugate back.

Algorithm 1 Random maximization

```
Input: Input $g$, backbone network $f$, scalar prior $h(x)$, sampler Sample_U() for $u \sim P$ sampled from a probability measure over $\mathcal{U}$ , number of random samples $K$, gradient descent step GD_Step

function Random_Maximization($g, f, h,$ Sample_U, $K$, use_GD, $T, \eta$, project_to_U)
    $\{u^1, \dots, u^K\} \leftarrow$ Sample_U($K$)
    $u^* \leftarrow \underset{\{u_1,\dots,u_K\}}{\arg\max} \; h \circ f(\kappa_{u^i}(g))$
    $u^* \leftarrow$ GD_Step($u^*$)

    return $u^*$
```

Take $f = \mathbb{1}_I$ for a Borel set $I \subset \mathbb{R}$. The indicator function $\mathbb{1}_I(\mathcal{L})$ is an orthogonal projection, since $\mathbb{1}_I(\mathcal{L})^2 = \mathbb{1}_I(\mathcal{L})$ and $\mathbb{1}_I(\mathcal{L})^* = \mathbb{1}_I(\mathcal{L})$.

Spectral Graph Filters. A *graph shift operator (GSO)* is a self-adjoint matrix that reflects the graph's connectivity, such as a (normalized) graph Laplacian or a symmetrized adjacency. Let $\mathcal{L}$ be such a GSO with eigenpairs $\{(\lambda_i, v_i)\}_{i=1}^N$ and $V = [v_1, \dots, v_N]$. For a T-channel node signal $\mathbf{X} \in \mathbb{R}^{N \times T}$ and a matrix-valued frequency response $g : \mathbb{R} \to \mathbb{R}^{d' \times T}$, the spectral filter

$$g(\mathcal{L})\mathbf{X} := \sum_{i=1}^{N} v_i v_i^\top \mathbf{X}\, g(\lambda_i)^\top$$

applies the graph convolution theorem (Bracewell & Kahn, 1966) with d' output channel: each Fourier mode v_i is preserved in space, while channels are mixed by $g(\lambda_i)$ in the spectral domain. In the scalar case ($T = d' = 1$) with $f : \mathbb{R} \to \mathbb{R}$, the spectral filter simply reduces to the functional-calculus operator acting on $\mathbf{X}$:

$$f(\mathcal{L})\mathbf{X} = \sum_{i=1}^{N} f(\lambda_i)\, v_i v_i^\top \mathbf{X} \;=\; V f(\Lambda) V^\top \mathbf{X}.$$

Spectral graph neural networks (Defferrard et al., 2016b; Kipf, 2016; Levie et al., 2018) compose such filters with pointwise nonlinearities, using trainable g at each layer.

E Application of Adaptive Canonicalization: Tutorial for Practitioners

In this section, we present the construction details of the application of adaptive canonicalization to anisotropic geometric networks. We start by describing the one vs. rest classifier and our maximization method.

E.1 One vs. Rest Classifiers

In our setting, each output channel $f_d \circ \rho_{f_d}^d(g) \in [0, 1]$ is a binary classifier, i.e., representing the probability of g being in class d vs. not being in class d (Rifkin & Klautau, 2004; Galar et al., 2011; Allwein et al., 2000). The per-class score is obtained by $\hat{y}_d = \sigma(f_d \circ \rho_{f_d}^d(g))$, where σ is a sigmoid function. Note that the vector $(\hat{y}_1, \dots, \hat{y}_D)$ is not a probability measure, since each entry represents the independent probability of class c being present. We use binary cross-entropy per class and sum over classes $\sum_{d=1}^{D}(-\, y_d \log \hat{y}_d - (1 - y_d) \log(1 - \hat{y}_d))$, where y_d denotes the true class.

E.2 Random Maximization

We estimate the prior maximization by searching over a transformation space (e.g., unitary orientations per spectral band or rotation for point clouds) and selecting the orientation that maximizes the chosen prior with the classification objective (i.e., one vs. rest). Specifically, we consider a probability measure on the space $\mathcal{U}_j$ and draw i.i.d samples $\{u_j^i\}_{i=1}^K$ from it. The argmax of $\{h_d \circ f_d\big(\kappa_{u_j}(g)\big)\}_{u_j \in \mathcal{U}_j}$ of

Eq. (2) is estimated as the argmax of

$$\{h_d \circ f_d(\kappa_{u_j^i}(g))\}_{i=1}^{K}.$$

For example, in anisotropic nonlinear spectral filters, we draw a finite pool of candidate transformations from the Haar measure on $\mathcal{U}_j$ (Mezzadri, 2006). For anisotropic point cloud networks, the search over rotations can be implemented with quaternion (Shoemake, 1992) or Rodrigues' formula (Rodrigues, 1840). For each input, we evaluate the prior objective for all candidates in the pool and keep only the maximizing orientation when computing the forward pass and gradients. In this way, the prior maximization is implemented as a randomized search: we sample a set of transformations, apply them in parallel, and pick the one giving the best prior value. Note that this sampling-based maximization is best thought of as one convenient implementation of our framework rather than a requirement, as other optimization strategies over $\mathcal{U}_j$ could be implemented and plugged as long as they approximately solve the same maximization problem.

To understand how well this random maximization approximates the ideal maximization over the full transformation space, we can adapt the analysis from Cordonnier et al. (2024). Their results provide a tail bound for approximating a maximum over a probability space by the maximum over random i.i.d. samples.

We first recall the notion of the volume retaining probability space introduced by Cordonnier et al. (2024).

Definition 21 (Volume retaining property (Cordonnier et al., 2024))**.** *Let $X \subset \mathbb{R}^d$ and let P be a probability measure on X. We say that the probability space (X, P) has the (r_0, κ)-volume retaining Lebesgue measure if there exist constants $r_0 > 0$ and $\kappa > 0$ such that for any $r \leq r_0$ and any $x \in X$*

$$P(B(x,r) \cap X) \geq \kappa \lambda_d(B(x,r)),$$

where λ_d is the d-dimensional Lebesgue measure and $B(x,r)$ is the ball center at x with radius r.

In our case, points are randomly sampled from some canonical measure over $\mathcal{U}_j$ (i.e., the Haar measure), and in all of our example applications $\mathcal{U}_j$ has the volume retaining property.

On a volume retaining space, Cordonnier et al. (2024) prove the following concentration inequality for maxima.

Lemma 22 (Concentration inequality for volume retaining space (Cordonnier et al., 2024))**.** *Let (X, P) be a probability space with the (r_0, κ)-volume retaining property and let $g : X^2 \to \mathbb{R}^q$ be K_g-Lipschitz. For any $\rho \geq \exp(-n\kappa r_0^d 2^d)$, for any random variables $X_1, \ldots, X_n \overset{i.i.d.}{\sim} P$, with probability at least $1 - \rho$, it holds*

$$\| \max_{1 \leq i \leq n} g(x, X_i) - \sup g(x, \cdot) \|_\infty \leq \frac{K_g}{2} \left(\frac{\ln(q/\rho)}{n\kappa} \right)^{1/d}.$$

Applying this lemma to our setting, g is the output of the task neural network on some class. Since typical neural networks are Lipschitz continuous (e.g., any multilayer perceptron based on a Lipschitz activation function), this immediately gives a guarantee that our random maximization method approximates the true maximum. We plan to extend this analysis for future work.

We note that prior maximization can be strengthened by locally refining the sample points u_j^i. Specifically, we initialize with random transformations drawn from the probability measure, perform the sampling-based prior maximization to select a candidate, and then run a few steps of gradient descent to further decrease the objective locally. If $\mathcal{U}$ is a manifold embedded in Euclidean space, one can apply the vanilla Euclidean gradient descent and then project the obtained transformation back to $\mathcal{U}$, or apply Riemannian gradient descent (Absil et al., 2008) to stay on $\mathcal{U}$. We summarize the random maximization in Alg. 1. In our experiments, we use 32 sampled transformations for anisotropic nonlinear spectral filters and use 50 sampled transformations for point clouds.

Finally, we note that when the prior maximization has error e (which is a random variable), this leads to an additive term e in universal approximation. Namely, for any $\epsilon > 0$, any continuous function can be approximated by $\theta \circ \rho_\theta$ up to error $\epsilon + e$ instead of ϵ in Theorem 6.

E.3 Construction Details for Anisotropic Nonlinear Spectral Filters

In spectral methods for graphs, we often use eigenvectors as a core component for graph representation learning. However, these eigenvectors are not uniquely defined. For each eigenvector we can flip its sign, and when an eigenvalue has multiplicity larger than one, any orthogonal basis of its eigenspace is valid. In practice, this means that the same graph can produce different eigenvectors depending on the eigensolver or numerical details, and a spectral graph neural network that takes these eigenvectors as input may give different outputs for the same graph, which affects stability and can reduce performance. Since the model should depend only on the graph structure and not on arbitrary choices of eigenvector bases, i.e., it should be invariant to the choice of the eigenbasis, it is important to remove such ambiguities. In this work, we focus on graph classification tasks with D classes and study how to resolve eigenbasis ambiguities in this setting.

To study this, we work in the following setup. Consider a graph $G = ([N], \mathbf{A}, \mathbf{S})$, where $[N]$ is the set of N vertices, $\mathbf{A} \in \mathbb{R}^{N\times N}$ is the adjacency matrix, and $\mathbf{S} \in \mathbb{R}^{N\times T}$ is an array of node features (row n us the T-dimensional feature at node n). We consider the normalized graph Laplacian $\mathcal{L} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$ as the graph shift operator (GSO)[2] in our experiments, where $\mathbf{D}$ is a diagonal degree matrix with diagonal entries $(d_i)_i \in \mathbb{R}^N$, where d_i is the degree of node i. The eigendecomposition of $\mathcal{L}$ is given by $\mathcal{L} = \mathbf{V}^{(G)}\mathbf{\Lambda}\mathbf{V}^{(G)\top}$, where $\mathbf{V}^{(G)} = (\mathbf{v}_i)_i \in \mathbb{R}^{N\times N}$ is an orthogonal matrix of eigenvectors as the columns (i.e., an eigenbasis) and $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \dots, \lambda_N)$ is the diagonal matrix of eigenvalues, where $0 \le \lambda_1 \le \ldots \le \lambda_N \le 2$. We then group the spectrum into predefined bands with boundaries $b_0 < b_1 < \cdots < b_B$ contained in $[0, 2]$, where $B \in \mathbb{N}$ is the total number of bands. The total band $[b_0, b_B]$ is a subset od $[0, 2]$ since the spectrum of the normalized Laplacian lies in this interval[3]. In our implementation, we use a dyadic partitioning scheme. Given a decay rate $0 < r < 1$, we set

$$b_0 = 0,\ b_k = 2r^{B-k} \text{ for } k = 1, \dots, B-1,\ b_B = 2,$$

and define the k-th band as $[b_{k-1}, b_k)$. A larger B yields narrower bands and hence finer spectral resolution. For example, taking $B = 5$ and $r = 0.5$ gives five bands: $[0, 0.125)$, $[0.125, 0.25)$, $[0.25, 0.5)$, $[0.5, 1)$ and $[1, 2)$.

We now make explicit the symmetry we want the model to respect. Recall that the eigendecomposition of the normalized Laplacian is $\mathcal{L} = \mathbf{V}^{(G)}\mathbf{\Lambda}\mathbf{V}^{(G)\top}$, where $\mathbf{V}^{(G)} = (\mathbf{v}_i)_i \in \mathbb{R}^{N\times N}$ collects the eigenvectors and $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \dots, \lambda_N)$ the eigenvalues. Given the band boundaries $b_0 < \cdots < b_B$, we define for each band k the index set $I_k(G) := \{i \in [N] : \lambda_i \in [b_{k-1}, b_k)\}$, and let $\mathbf{V}_k^{(G)} \in \mathbb{R}^{N\times M_k(G)}$ be the submatrix of $\mathbf{V}^{(G)}$ whose columns are the eigenvectors $(\mathbf{v}_i)_{i\in I_k(G)}$. Therefore, we can write $\mathbf{V}^{(G)} = [\mathbf{V}_1^{(G)} | \cdots | \mathbf{V}_B^{(G)}]$. For each band we denote the associated Paley-Wiener space by $\mathcal{X}_k(G) := \mathrm{span}\{\mathbf{v}_i : i \in I_k(G)\} = \mathrm{Im}(\mathbf{V}_k^{(G)})$. The ambiguity we want to handle comes from changing the orthonormal basis inside each band. Any other basis that of the k-th band has the form $\widetilde{\mathbf{V}}_k^{(G)} = \mathbf{V}_k^{(G)}\mathbf{U}_k^{(G)}$, where $\mathbf{U}_k^{(G)} \in \mathbb{R}^{M_k(G)\times M_k(G)}$ is a unitary matrix. Equivalently, we can write a full orthonormal basis of $\mathbb{R}^N$ whose columns are partitioned into subsequences corresponding to the bands, and each subsequence spans the associated Paley-Wiener space $\mathcal{X}_k(G)$. We can write this condition in matrix form as: $\mathbb{R}^{N\times N} \ni \widetilde{\mathbf{V}}^{(G)} = \mathbf{V}^{(G)}\mathbf{U}^{(G)}$, where $\mathbf{U}^{(G)} := \mathrm{diag}(\mathbf{U}_1^{(G)}, \dots, \mathbf{U}_B^{(G)}) \in \mathbb{R}^{N\times N}$, i.e., the a block matrix with diagonal blocks $\mathbf{U}_k^{(G)}$. The set of block-diagonal unitary matrices $\mathcal{U}^{(G)} = \{\mathbf{U}^{(G)} = \mathrm{diag}(\mathbf{U}_1^{(G)}, \dots, \mathbf{U}_B^{(G)})\}$ constitutes the band-preserving transformations, i.e., the unitary matrices that keep the Paley-Wiener spaces (or bands) invariant in the sense that $\mathbf{U}^{(G)}\mathbf{v} \in \mathcal{X}_k(G), \forall\, k \in [B],\ \forall\, \mathbf{v} \in \mathcal{X}_k(G)$. The space of these matrices constitute the symmetry space in our setting.

The above construction is defined for a single graph, but in our setting we work with a collection of graphs. The spectral support of the bands, that is, the intervals $[b_{k-1}, b_k)$, is fixed across all graphs. However, for a given graph G, the k-th band, i.e, the linear span of the eigenvectors whose eigenvalues lie in $[b_{k-1}, b_k)$, and this subspace depends on G. Different graphs can have a different number of eigenvalues in the same interval, so the dimension of the k-th band is not constant across graphs. We denote by $M_k(G)$ the dimension of the k-th band for the graph G.

[2] Note that any other self-adjoint GSO can be chosen.

[3] For other GSOs, different bands can be chosen to match their spectral range, or to cover a subset of the spectrum.

In order to apply a task network given by a standard neural network architecture, such as an MLP, we require that all inputs lie in a common vector space of fixed dimension, independent of the particular graph G. The variability of the band dimensions $M_k(G)$ across graphs violates this requirement. To resolve this, we introduce a padding operator that standardizes the size of each band. Implementation-wise, for each band k we fix for each band k the size to be the largest value of $M_k(G)$ over the dataset and denote this value by J_k, and we zero-pad bands that are smaller than this size. Let $J_1, \dots, J_B$ be these integers, and denote $J = \sum_{k=1}^{B} J_k$. For a graph G with signal $\mathbf{S} \in \mathbb{R}^{N\times T}$, let $C_k(G, \mathbf{S}) \in \mathbb{R}^{M_k(G)\times T}$ be the matrix of spectral coefficients in the k-th band (one row per eigenvector in the band and T channels). Since $M_k(G) \leq J_k$ by construction, we define the band-wise padding operator P_k by

$$P_k(G, \mathbf{S}) \coloneqq \begin{bmatrix} C_k(G, \mathbf{S}) \\ \mathbf{0}_{(J_k - M_k(G))\times T} \end{bmatrix} \in \mathbb{R}^{J_k \times T}.$$

Collecting all bands, we obtain the global operator $P(G, \mathbf{S}) \coloneqq (P_k(G, \mathbf{S}))_{k=1}^{B} \in \mathbb{R}^{J\times T}$. By construction, P maps every graph–signal pair $(G, \mathbf{S})$ to a spectral representation in the fixed-dimensional vector space $\mathbb{R}^{J\times T}$, so that we can apply the same task network $\Psi : \mathbb{R}^{J\times T} \to \mathbb{R}^{D}$ (e.g., an MLP) uniformly across all graphs. After standardizing the band sizes to J_k and writing $J = \sum_{k=1}^{B} J_k$, we use the same parameterization with $\{\mathbf{U} = \mathrm{diag}(\mathbf{U}_1, \dots, \mathbf{U}_B) \in \mathbb{R}^{J\times J}$, where $\mathbf{U}_k \in \mathbb{R}^{J_k \times J_k}$ is unitary$\}$, which gives a parameterization of the space of unitary operators that keep the bands invariant.

To resolve the band-wise eigenbasis ambiguity, we introduce the anisotropic nonlinear spectral filters as follows. Consider a symmetryless neural network $\Psi : \mathbb{R}^{J\times T} \to \mathbb{R}^{D}$ that operates on the space of spectral coefficients of the signal. We denote $\Psi = (\Psi_d)_{d=1}^{D}$ where $\Psi_d : \mathbb{R}^{J\times T} \to \mathbb{R}$. The prior maximization for each class $d \in \{1, \dots, D\}$ is performed by

$$\{\mathbf{U}_1^{\square(d)}, \dots, \mathbf{U}_B^{\square(d)}\} = \operatorname*{arg\,max}_{\mathbf{U}_1^{(d)}, \dots, \mathbf{U}_B^{(d)}} \; h_d(\Psi_d(\mathbf{U}_1^{(d)}(G, \mathbf{S})), \dots, \mathbf{U}_B^{(d)}(P_B(G, \mathbf{S})))),$$

where $\mathbf{U}_k^{(d)} \in \mathbb{R}^{J_k \times J_k}$ is a unitary matrix and $h_d(x) = x$ is class-d prior. Once we obtain the set of the optimal unitary matrices $\{\mathbf{U}_1^{\square(d)}, \dots, \mathbf{U}_B^{\square(d)}\}$, the class d score is then computed by $\Psi_d(\mathbf{U}_1^{\square(d)}(P_1(\mathbf{V}^{(G)}, \mathbf{S})), \dots, \mathbf{U}_B^{\square(d)}(P_B(\mathbf{V}^{(G)}, \mathbf{S})))$ are then passed through a sigmoid nonlinearity to obtain class probabilities. The training loss is the sum of D binary cross-entropies.

E.4 Construction Details for Anisotropic Point Cloud Networks

We next apply adaptive canonicalization in the spatial domain, where inputs are multisets of points in $\mathbb{R}^3$ with (x, y, z) coordinates. Specifically, we focus on multiset networks applied to classification tasks. Standard multiset networks, e.g., DeepSet (Zaheer et al., 2017) and PointNet (Qi et al., 2017a), which take the 3D coordinates of the points as their features, suffer from an inherent anisotropy problem (Wiersma et al., 2022; Qian et al., 2021): the classification outcomes for a point cloud and for a rotated version of the same point cloud do not coincide in general. To address this sensitivity to orientation in multiset networks, we introduce an adaptive canonicalization module that searches over the 3D rotation group $\mathcal{SO}(3)$.

Let $\mathbf{X} \in \mathbb{R}^{N\times 3}$ represent a point cloud of N points in $\mathbb{R}^3$ and let $\Psi : \mathbb{R}^{N\times 3} \to \mathbb{R}^{D}$ be a multiset neural network. In classification, D is the number of classes. We denote $\Psi = (\Psi_d)_{d=1}^{D}$ where $\Psi_d : \mathbb{R}^{N\times 3} \to \mathbb{R}$. Here, Ψ is invariant to permutations of the N points, but not invariant to 3D rotations of the point cloud. The prior maximization for each class $d \in \{1, \dots, D\}$ is performed by

$$\mathbf{R}_d^{\square} = \operatorname*{arg\,max}_{\mathbf{R}\in\mathcal{SO}(3)} h_d(\Psi_d(\mathbf{X}\mathbf{R}^{\top})),$$

where class-d prior is $h_d(x) = x$. Once we found the canonical form $\mathbf{R}_d^{\square}$, the class scores in our anisotropic point cloud network are computed as $s_d = \Psi_d(\mathbf{X}\mathbf{R}_d^{\square\top})$ and are then passed through a sigmoid nonlinearity to obtain class probabilities. The training loss is the sum of D binary cross-entropies.

The canonicalization setting is independent of the specific multiset neural network architecture. Below, we present three widely used multiset neural networks equipped with adaptive canonicalization.

Adaptive Canonicalization Applied to DeepSet (AC-DeepSet). We begin by briefly recalling DeepSet (Zaheer et al., 2017), which is a general framework for learning functions on sets and has UAT (Wagstaff et al., 2022). Given a point cloud represented as $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \subset \mathbb{R}^3$, DeepSet computes a permutation-invariant representation by encoding each point with an MLP ϕ, aggregating features with summation, and applying a global function ξ:

$$\mathbf{x}' = \xi\left(\sum_{i=1}^{N} \phi(\mathbf{x}_i)\right).$$

To apply adaptive canonicalization to DeepSet, we define, for each class $d \in \{1, \ldots, D\}$, a class-dependent canonical rotation by

$$\mathbf{R}_d^{\square} = \arg\max_{\mathbf{R} \in \mathcal{SO}(3)} h_d\left(\Psi_d\left(\xi\left(\sum_{i=1}^{N} \phi(\mathbf{x}_i \mathbf{R}^\top)\right)\right)\right),$$

where Ψ_d is the one-vs-rest classifier for class d, and $h_d(x) = x$ is the prior. Once $\mathbf{R}_d^{\square}$ is obtained, the class scores are computed as $s_d = \Psi_d(\xi(\sum_{i=1}^{N} \phi(\mathbf{x}_i \mathbf{R}_d^{\square\top})))$ for all $d \in \{1, \ldots, D\}$, and passed through sigmoids to obtain class-wise probabilities. The network is then trained using the sum of D binary cross-entropy losses.

Adaptive Canonicalization Applied to PointNet (AC-PointNet). Given a point cloud $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \in \mathbb{R}^3$, PointNet (Qi et al., 2017a) processes each point independently with a shared MLP ϕ, and then aggregates points features using a symmetric function (e.g., max-pooling)

$$\mathbf{x}' = \mathrm{Pool}(\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_N)).$$

To equip PointNet with adaptive canonicalization, we define a class-specific orientation $\mathbf{R}_d^{\square}$ for each class d obtained by

$$\mathbf{R}_d^{\square} = \arg\max_{\mathbf{R} \in \mathcal{SO}(3)} h_d(\Psi_d(\mathrm{Pool}(\phi(\mathbf{x}_1 \mathbf{R}^\top), \ldots, \phi(\mathbf{x}_N \mathbf{R}^\top)))),$$

where $h_d(x) = x$ is the class-d prior and Ψ_d is the class-specific classifier conducting binary classification in the one vs. rest manner. Once having $\mathbf{R}_d^{\square}$, the scoring head is denoted by $s_d = \Psi_d(\mathrm{Pool}(\phi(\mathbf{x}_1 \mathbf{R}_d^{\square\top}), \ldots, \phi(\mathbf{x}_N \mathbf{R}_d^{\square\top})))$. Then we define the per-class head score by a sigmoid and use binary cross-entropy per class and sum over classes as the training loss.

Adaptive Canonicalization Applied to DGCNN (AC-DGCNN). Consider the input $\mathbf{X} \in \mathbb{R}^{N \times 3}$ as a point cloud with points $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \in \mathbb{R}^3$. DGCNN (Wang et al., 2019) constructs dynamic k-nearest graphs by computing $G = (V, E)$ where $E = \{(i, j) : j \in \mathrm{kNN}(\mathbf{x}_i, k)\}$. Then, the edge convolution is performed by computing the edge features and applying a max pooling:

$$\mathbf{x}_i' = \mathrm{Pool}_{(i,j) \in E}(\mathrm{ReLU}(\Psi(\mathbf{x}_j - \mathbf{x}_i, \mathbf{x}_i))).$$

Applying adaptive canonicalization to the DGCNN architecture, we define a class-specific orientation $\mathbf{R}_d^{\square}$ obtained by optimizing a class-dependent prior $h_d(x) = x$:

$$\mathbf{R}_d^{\square} = \arg\max_{\mathbf{R} \in \mathcal{SO}(3)} h_d(\Psi_d(\mathrm{Pool}_{(i,j) \in E}(\mathrm{ReLU}(\Psi((\mathbf{x}_j - \mathbf{x}_i)\mathbf{R}^\top, \mathbf{x}_i \mathbf{R}^\top))))), \tag{5}$$

where Ψ_d is the class-d classifier. Then, the classifier is applied to the output under $\mathbf{R}_d^{\square}$ for computing $s_d = \Psi_d(\mathrm{Pool}_{(i,j) \in E}(\mathrm{ReLU}(\Psi((\mathbf{x}_j - \mathbf{x}_i)\mathbf{R}^{\square\top}, \mathbf{x}_i \mathbf{R}^{\square\top}))))$. The per-class head score is then computed by a sigmoid, and the sum over D binary cross-entropies is used as the training loss.

E.5 Truncation Canonicalization

We introduce truncation-based prior maximization on images as an additional application of our adaptive canonicalization framework. Intuitively, many image classes are invariant under removing uninformative regions: as long as the object of interest remains in the field of view, the class label should not change. We consider, in this case, the truncation as our "symmetry" and apply prior maximization over this family of transformations to select a canonical truncation. We note that this transformation family is not based on a group action.

We model images as elements of a set of $\mathcal{G}$ in the following way. Given $N \in \mathbb{N}$, consider the regular grid on the unit square $[0,1]^2$ given by the pixels $Q_{i,j} = [\frac{i}{N}, \frac{i+1}{N}) \times [\frac{j}{N}, \frac{j+1}{N})$ for $0 \leq i, j < N$. We define $\mathcal{G}$ to be the set of all functions $g \in L_2(\mathbb{R}^2)$ such that $\mathrm{supp}(g) \subset [0,1]^2$ and g is piecewise constant on the grid, i.e., there exist coefficients $v_{i,j} \in [0,1]$ such that $g(x) = v_{i,j}$ for all $x \in \{Q_{i,j}\}$. Since each image $g \in \mathcal{G}$ is uniquely determined by its collection of pixel values $(v_{i,j})_{0 \leq i,j < N} \in [0,1]^{N^2}$, we can identify (via an isometric isomorphism) $\mathcal{G}$ with with the box $[0,1]^{N^2} \subset \mathbb{R}^{N^2}$. Since closed and bounded subsets of finite-dimensional Euclidean spaces are compact, the box $[0,1]^{N^2} \subset \mathbb{R}^{N^2}$ is compact. Because $\mathcal{G}$ is isometrically isomorphic to $[0,1]^{N^2}$, the set $\mathcal{G}$ is also compact (and therefore locally compact).

The truncation is parameterized by four coordinates collected in a vector $u = (x_{\mathrm{L}}, y_{\mathrm{T}}, x_{\mathrm{R}}, y_{\mathrm{B}}) \in \mathcal{U} \subset [0,1]^4$, where we impose the constraints $0 \leq x_{\mathrm{L}} \leq x_{\mathrm{R}} \leq 1$ and $0 \leq y_{\mathrm{T}} \leq y_{\mathrm{B}} \leq 1$. We consider the standard Euclidean metric in $\mathcal{U}$. Since $\mathcal{U}$ is a closed and bounded subset of $[0,1]^4$, it is compact.

We now define the truncation transformation. For a parameter $u = (x_{\mathrm{L}}, y_{\mathrm{T}}, x_{\mathrm{R}}, y_{\mathrm{B}}) \in \mathcal{U}$, we view u as encoding the top-left and bottom-right corners of an axis-aligned rectangle: $R(u) = [x_{\mathrm{L}}, x_{\mathrm{R}}] \times [y_{\mathrm{T}}, y_{\mathrm{B}}] \subset [0,1]^2$. As u varies in the compact box $\mathcal{U}$, this truncation window moves continuously inside the domain containing all images. Given an image $g \in \mathcal{G}$, we define the truncation by

$$x \mapsto \begin{cases} g(x), & x \in R(u), \\ 0, & x \notin R(u). \end{cases}$$

After truncation, the function lies in $L_2(\mathbb{R}^2)$ but not in $\mathcal{G}$ in general. We then project it back onto the finite-dimensional subspace $\mathcal{G}$ of piecewise constant functions by the orthogonal projection $P : L_2(\mathbb{R}^2) \to \mathcal{G}$ defined by

$$(Pg)(x) = \sum_{i,j=1}^{N} \bar{g}_{ij}\, \mathbb{1}_{Q_{ij}}(x), \qquad \bar{g}_{ij} = N^2 \int_{Q_{ij}} g(y)dy.$$

We then define $\kappa_u(g) := P(g\mathbb{1}_{R(u)}) \in \mathcal{G}$.

We now verify that this truncation family satisfies our assumptions. By construction, $\mathcal{U}$ is compact, and the image space $\mathcal{G}$ is compact. In our setting we have $\mathcal{K} = \mathcal{G}$, so $\mathcal{K}$ is also compact. The map

$$\kappa_{(\cdot)}(\cdot\cdot) : \mathcal{U} \times \mathcal{G} \to \mathcal{K}, \quad (u, g) \mapsto \kappa_u(g) \in \mathcal{K}$$

is continuous, and hence, by compactness of the product $\mathcal{U} \times \mathcal{G}$, it is uniformly continuous. This implies that $\{g \mapsto f(\kappa_u(g))\}_{u \in \mathcal{U}}$ is equicontinuous. Thus, the truncation transformation family satisfies the conditions for continuous canonicalization.

We focus on the image classification task with D classes and take axis-aligned truncations as the transformation family. Let $\mathbf{X} \in \mathbb{R}^{N \times N}$ be an image. A truncation is specified by a side-length $s \in [s_{\text{min}}, s_{\text{max}}] \subset [0.5, 1.0]$ and a discrete top-left corner. For a given s, we set the truncation side length to $M(s) = \lfloor sN \rfloor$, and require the window to lie inside the image domain, i.e., $1 \leq i_0 \leq N - M(s) + 1$ and $1 \leq j_0 \leq N - M(s) + 1$. The corresponding axis-aligned window is $\{(i,j) \in \{1, \dots, N\}^2 \,|\, i_0 \leq i \leq i_0 + M(s) - 1,\ j_0 \leq j \leq j_0 + M(s) - 1\}$. The truncation then keeps only the pixels inside the window and zeros out the rest, which yields a truncated patch $C_{(i_0,j_0,s)}(\mathbf{X}) \in \mathbb{R}^{M(s) \times M(s)}$. To keep the input dimension fixed, we then rescale this cropped patch back to the original resolution $N \times N$ using a standard interpolation scheme. We denote this resizing operator by $R_s : \mathbb{R}^{M(s) \times M(s)} \to \mathbb{R}^{N \times N}$. The truncation operator is therefore defined as the composition $T_{(i_0,j_0,s)}(\mathbf{X}) = R_s(C_{(i_0,j_0,s)}(\mathbf{X})) \in \mathbb{R}^{N \times N}$. The transformation family used in our method is

$$\mathcal{T} = \{T_{(i_0,j_0,s)} \mid s \in [s_{\text{min}}, s_{\text{max}}],\ (i_0, j_0) \text{ admissible as above}\},$$

and in practice random truncations are obtained by sampling s and (i_0, j_0) from a suitable distribution under these constraints.

We introduce the truncation canonicalization as follows. Consider a symmetryless neural network $\Psi : \mathbb{R}^{N \times N} \to \mathbb{R}^D$ that operates on the images. We denote $\Psi = (\Psi_d)_{d=1}^{D}$ where $\Psi_d : \mathbb{R}^{N \times N} \to \mathbb{R}$. The prior maximization for each class $d \in \{1, \dots, D\}$ is performed by

$$T^{\square}_{(i_0,j_0,s)} = \underset{T_{(i_0,j_0,s)} \in \mathcal{T}}{\arg\max}\ h_d(\Psi_d(T_{(i_0,j_0,s)}(\mathbf{X}))),$$

where $h(x) = x$. Once we obtain $T^{\square}_{(i_0,j_0,s)}$, the class d score is then computed by $\Psi_d(T^{\square}_{(i_0,j_0,s)}(\mathbf{X}))$ are then passed through a sigmoid nonlinearity to obtain class probabilities. The training loss is the sum of D binary cross-entropies.

F Experimental Details

In this section, we describe the experimental setups and implementation details used in Sec. 5.

F.1 Illustrative Toy Problems: Grid Signal Orientation Tasks

Toy Problem and Experimental Setup. We consider square grid on the torus with a 2-channel signal. The first channel contains a sinusoidal signal aligned with the x-axis, given by $\sin(2\pi x/T)$. The second channel depends on the class label: in Class 0 it is aligned with the x-axis, while in Class 1 it encodes a sinusoidal signal along the y-axis, $\sin(2\pi y/T)$. Independent Gaussian noise with variance σ^2 is added to each channel. In addition, it introduces an additional challenge by spatially restricting the support of the channels. The grid is vertically partitioned into two disjoint halves. The first channel is supported only on the left half. The second channel is supported only on the right half. The task is to decide if the frequency at the two channels is in the same orientation. The grid size is fixed at $N = 40^2$, the sinusoidal period is set to $T = 20$, and the noise level is chosen as $\sigma = 0.1$. We generate 1000 samples. Evaluation is carried out using 10-fold cross-validation.

Competing Methods. The competing methods include: MLP, GCN, GAT, GIN, ChebNet, NLSF, S^{2}GNN, FA+GIN, and OAP+GIN.

Hyperparameters. We use a three-layer network with a hidden feature dimension chosen from $\{32, 64, 128\}$ and ReLU activation functions. The learning rate is selected from $\{10^{-3}, 10^{-4}, 10^{-5}\}$. Batch size 100. All models are implemented in PyTorch and optimized with the Adam optimizer (Kingma & Ba, 2014). Experiments are conducted on an Nvidia DGX A100. The output of the GNN is then passed to an MLP, followed by a softmax classifier.

F.2 Graph Classification on TUDataset

Datasets and Experimental Setup. We consider five graph classification benchmarks from TUDataset (Morris et al., 2020): MUTAG, PTC, ENZYMES, PROTEINS, and NCI1. The dataset statistic is reported in Tab. 4. Following the random split protocol (Ma et al., 2019; Ying et al., 2018; Zhang et al., 2019b), we partition the dataset into 80% training, 10% validation, and 10% testing. Results are averaged over 10 random splits, with mean accuracy and standard deviation reported.

Competing Baselines. We evaluate on medium-scale graph classification benchmarks from TUDataset, using the same set of competing methods as in grid signal orientation tasks. The baselines include MLP, GCN, GAT, GIN, ChebNet, NLSF, S^{2}GNN, FA+GIN, and OAP+GIN.

Hyperparameters. The hidden dimension is set to be 128. The models are implemented using PyTorch, optimized with the Adam optimizer (Kingma & Ba, 2014). An early stopping strategy is applied, where training halts if the validation loss does not improve for 100 consecutive epochs. The hyperparameters are selected through a grid search, conducted via Optuna (Akiba et al., 2019), with with the learning rate and weight decay explored in the set $\{1e^{-2}, 1e^{-3}, 1e^{-4}\}$, the pooling ratio varying within $[0.1, 0.9]$ with step 0.1, and the number of layers ranging from 2 to 9 in a step size of 1. The output representations are then passed into an MLP followed by a softmax layer, and predictions are obtained by optimizing a cross-entropy loss function. Experiments are conducted on an Nvidia DGX A100.

F.3 Molecular Classification on OGB Datasets

Datasets and Experimental Setup. We evaluate on larger-scale benchmarks from the Open Graph Benchmark (OGB) dataset (Hu et al., 2020) for classification tasks, including ogbg-molhiv, ogbg-

Table 4: Datasets statistics.

Dataset	# Graphs	# Classes	Avg.# Nodes	Avg.# Edges
MUTAG	188	2	17.93	19.79
PTC	344	2	14.29	14.69
ENZYMES	600	6	32.63	64.14
PROTEINS	1113	2	39.06	72.82
NCI1	4110	2	29.87	32.30
ogbg-molhiv	41127	2	25.5	27.5
ogbg-molpcba	437929	128	26.0	28.1
ogbg-ppa	158100	37	243.4	2266.1

molpcba, and ogbg-ppa. Dataset statistics are summarized in Tab. 4 The evaluation settings are followed by the OGB protocol (Hu et al., 2020).

Competing Baselines. For large-scale graph classification, we include GCN, GIN, GatedGCN, PNA, GraphTrans, SAT, GPS, SAN, and the canonicalization-based variant OAP+GatedGCN as the competing methods. These approaches have previously demonstrated strong performance on OGB benchmarks, and their reported results are taken from prior work[4].

Hyperparameters. The models are implemented in PyTorch and optimized with the Adam optimizer, with training capped at a maximum of 1000 epochs and controlled by an early stopping criterion. The hidden dimension is selected from the set $\{128, 256, 512\}$, while the number of layers varies from 2 to 10 in steps of 1. Dropout rates are explored within the range $[0, 0.1, \ldots, 0.5]$, the learning rate is tuned within the interval $[0.0001, 0.001]$, and the warmup is set as 5 or 10. Additionally, the batch size is chosen from $\{32, 64, 128, 256\}$ and the weight decay is chosen from $\{10^{-4}, 10^{-5}, 10^{-6}\}$. All hyperparameters are tuned using Optuna (Akiba et al., 2019). The experiments are conducted on an NVIDIA A100 GPU.

F.4 ModelNet40 Point Cloud Classification

Datasets and Experimental Setup. Our evaluation for point cloud classification was carried out on the ModelNet40 dataset (Wu et al., 2015), which consists of 40 object categories and a total of 12,311 3D models. Following prior studies (Wang et al., 2019; Deng et al., 2021), we allocated 9,843 models for training and 2,468 models for testing in the classification task. For each model, 1,024 points were uniformly sampled from its mesh surface, using only the xyz coordinates of the sampled points. We apply on-the-fly rotation augmentation during training, following Esteves et al. (2018); Deng et al. (2021), such that the dataset size remains unchanged. At test time, each example is rotated by an arbitrary rotation. Note that on-the-fly augmentation essentially changes the training data distribution during the learning process. The purpose of comparing under rotation protocols is to assess a model's invariance to rotational changes.

Competing Baselines. For point cloud classification tasks, we compare our anisotropic geometric method with point cloud approaches, including PointNet and DGCNN architectures, as well as equivariant models based on the vector neuron framework, i.e., VN-PointNet and VN-DGCNN. We further include canonicalization baselines, CN-PointNet and CN-DGCNN, and traditional augmentation baselines in which the training set is expanded with pre-generated rotations (PointNet-Aug, DGCNN-Aug) with a factor of five ($\times 5$). The experimental results of PointNet, DGCNN, VN-PointNet, and VN-DGCNN are taken from Wang et al. (2019); Deng et al. (2021).

Hyperparameters. We follow the published hyperparameters and training protocol of PointNet and DGCNN. For PointNet, we uses identical channel widths to PointNet (64, 64, 64, 128, 1024). We use Adam optimizer with learning rate 0.001 and batch size 32 with a weight decay 1×10^{-4} and dropout 0.3. For DGCNN, each input comprises 1,024 uniformly sampled points, and the k-NN graph uses neighborhood size $k = 20$. DGCNN uses four EdgeConv layers (with per-layer MLPs of

[4] https://ogb.stanford.edu/docs/leader_graphprop/

sizes 64, 64, 128, 256). We train with stochastic gradient descent (initial learning rate 0.1) and apply a cosine annealing schedule of 0.001. Training runs for 250 epochs with a batch size of 32, and we use a dropout rate of 0.5 in the fully connected layers.

G Additional Results

In this section, we present additional results of our adaptive canonicalization, including experimental trade-offs of sampling-based and optimization-based construction, anisotropic nonlinear spectral filters for node-level representation, and out-of-sample rotation generalization for point clouds.

G.1 Sampling-based VS Optimization-based Implementation

To evaluate the trade-offs between sampling-based and our sample-and-refine (optimization-based) implementation, we conduct experiments on the TUDataset graph classification benchmarks. Tab. 5 reports the classification performance of TUDaset under sampling-based and optimization-based adaptive caninocalization. We see that the optimization-based implementation consistently performs better than the sampling-based one. While increasing the sampling candidates (from $1\times$ to $5\times$ or $10\times$) improves the performance, the sample-and-refine strategy is more memory-efficient than massive sampling approaches. Rather than storing and evaluating hundreds of rotation matrices simultaneously, it processes a smaller working set through iterative refinement, reducing memory pressure (Li et al., 2022). In terms of computation time, the sampling-based method grows linearly with the number of candidates, while the optimization-based method add a small overhead to the inner steps. We see that in practice, a modest refinement (a few steps) surpasses the accuracy of large sampling budgets at a lower time, offering a better accuracy-time trade-off.

Table 5: Comparison of sampling-based vs. optimization-based adaptive canonicalization. Classification accuracy across TUDataset. Sampling methods use different numbers of random candidates, while the optimization approach combines sampling with local refinement via gradient descent.

	MUTAG	PTC	ENZYMES	PROTEINS	NCI1
A-NLSF (sampling)	84.23±1.4	69.05±1.8	70.10±1.5	82.94±1.6	80.64±1.2
A-NLSF (sampling ×5)	85.17±1.3	72.21±1.3	71.59±1.0	83.57±1.8	80.92±1.3
A-NLSF (sampling ×10)	85.54±1.3	72.78±1.5	72.42±1.2	85.03±1.2	80.94±0.8
A-NLSF (optimization)	87.94±0.9	73.16±1.2	73.01±0.8	85.47±0.6	82.01±0.9

G.2 Node-Level Anisotropic Nonlinear Filters

We introduce the adaptive canonicalization applied to spectral graph neural networks for learning graph-level representation in Sec. 4.1 and App. E.3. The adaptive canonicalization can also be applied to node-level representation, where the node-level representation proceeds by mapping the input signal to the spectral domain (Mallat, 2002) in a band-wise manner with an oriented basis within each band's eigenspace, and performing a synthesis step that transforms the learned coefficients back to the node domain.

On a square grid, each x Fourier mode has a corresponding y Fourier mode of the same response. Therefore, standard spectral methods are inherently isotropic as they cannot distinguish between horizontal and vertical directional information. On the other hand, adaptive canonicalization is anisotropic and our method can learn distinct orientations. The resulting spatial operator can therefore implement any directional filter that a convolutional neural network can achieve (Shannon, 2006; LeCun & Bengio, 1998; Freeman et al., 1991; Dagès et al., 2024).

In graph-level tasks, the canonicalized node-level embeddings can serve as the intermediate representation from which graph-level features are derived. Specifically, the resulting node embeddings can be aggregated through standard pooling operations to have a graph-level representation. We evaluate the node-to-graph construction on TUDataset for graph classification tasks. The results are summarized in Tab. 6. We see that the node-to-graph construction achieves performance closely aligned with, and in some cases approaching, that of the direct graph-level canonicalization. We attribute the slightly worse performance to the potential pooling loss.

Table 6: Graph classification performance on TUDataset using adaptive canonicalization. Comparison between direct graph-level representations (Graph) and node-to-graph constructions (Node-to-graph).

	MUTAG	PTC	ENZYMES	PROTEINS	NCI1
Node-to-graph	87.02±1.1	72.14±1.5	71.26±1.2	84.87±0.8	81.64±1.2
Graph	87.94±0.9	73.16±1.2	73.01±0.8	85.47±0.6	82.01±0.9

G.3 Out-of-Sample Rotation Generalization in Point Clouds

We adopt the $z/\mathcal{SO}(3)$ protocol (Esteves et al., 2018; Deng et al., 2021): training with on-the-fly azimuthal rotations (z-axis) augmentation, and evaluation applies under arbitrary rotations to each test shape. In this setting, we assess out-of-sample rotation generalization by constraining training data rotations while testing on the full rotation group. The classification performance on ModelNet40 under $z/\mathcal{SO}(3)$ protocol is reported in Tab. 7. Standard PointNet and DGCNN drop sharply under this shift. Equivariant vector-neuron variants recover much of the loss, and canonicalization baselines are comparable. Our adaptive canonicalization outperforms both equivariant architecture and canonicalization baselines in both backbones.

Table 7: Classification accuracy on ModelNet40 under $z/\mathcal{SO}(3)$ protocol.

	PointNet	DGCNN	VN-PointNet	VN-DGCNN	CN-PointNet	CN-DGCNN	AC-PointNet	AC-DGCNN
Accuracy	19.6	33.8	77.5	89.5	79.6	88.8	81.4	91.8

G.4 Ablation Studies

We conduct ablation studies on the spectral band partitioning and the choice of GSO for A-NLSF, as well as on the impact of different point cloud backbones in our anisotropic point cloud networks.

G.4.1 Spectral Band Partition

In our experiment, we adopt a dyadic partitioning scheme (see App. E.3). In this section, we conduct an ablation using a uniform partitioning of the eigenvalues with the same number of bands and report the graph classification performance in Tab. 8. We see that using the dyadic partitions performs better than using the uniform partition. tion provided by dyadic bands, which could more effectively isolate band-wise unitary actions that commute with the chosen GSO. We also note that spectral band design can be realized in more flexible and expressive ways, for example, through attention as in SpecFormer (Bo et al., 2023). Investigating such learned or adaptive band-selection strategies is an important direction for future work and may further strengthen our adaptive canonicalization framework.

Table 8: Graph classification performance using uniform and dyadic spectral band partitioning.

	MUTAG	PTC	ENZYMES	PROTEINS	NCI1
Uniform	81.36±1.2	66.20±0.8	62.84±1.4	80.01±1.3	79.62±1.0
Dyadic	87.94±0.9	73.16±1.2	73.01±0.8	85.47±0.6	82.01±0.9

G.4.2 Graph Shift Operator

We evaluate the graph Laplacian as an alternative GSO. Tab. 9 reports the graph classification performance of A-NLSF when instantiated with the graph Laplacian versus the normalized graph Laplacian. We observe that using the normalized graph Laplacian in our method yields better performance than the graph Laplacian. We attribute this to the properties of the normalized Laplacian: (i) the normalized Laplacian removes degree-related scaling effects, leading to a comparable spectral domain across graphs with different degree distributions, and (ii) its eigenvalues lie in the fixed interval $[0, 2]$, providing a controlled and interpretable frequency range, and making dyadic partitioning better aligned across different graphs.

G.4.3 Point cloud backbones

Table 9: Graph classification performance of A-NLSF with different GSO.

	MUTAG	PTC	ENZYMES	PROTEINS	NCI1
Graph Laplacian	83.76±1.0	67.23±1.4	62.60±1.2	82.64±1.6	78.59±0.8
Normalized graph Laplacian	87.94±0.9	73.16±1.2	73.01±0.8	85.47±0.6	82.01±0.9

In order to assess the impact of the backbone choice on the performance of our anisotropic point cloud networks, we extended our experiments to include two additional and widely used point cloud backbones, PointNet++ (Qi et al., 2017b) and RSCNN (Liu et al., 2019), in addition to PointNet and DGCNN reported in Tab. 3. We denote the corresponding variants by AC-PointNet++ and AC-RSCNN. The ablation results are reported in Tab. 10. We see that the choice of backbone does influence the overall point cloud classification performance. However, we observe that our adaptive canonicalization framework consistently improves the classification performance across these backbones. Moreover, when comparing methods built on the same backbone (e.g., PointNet or DGCNN), our approach outperforms equivariant models, data augmentation, and standard canonicalization (see Tab. 3). This indicates that our method is robust across different point cloud backbones and can further benefit from stronger backbones when they are available.

Table 10: Classification results on ModelNet40 for different point cloud backbones. Results of competing methods marked with * are taken from Deng et al. (2021); Luo et al. (2022).

	Accuracy
PointNet	74.7*
AC-PointNet	81.1±0.7
PointNet++	85.0*
AC-PointNet++	87.4±0.4
RSCNN	82.6*
AC-RSCNN	87.6±0.3
DGCNN	88.6*
AC-DGCNN	91.6±0.6

G.5 Sensitivity Analysis

To examine the effect of different hyperparameters, we conduct a hyperparameter sensitivity study covering grid size, sinusoidal period, noise level, and hidden dimension. For each hyperparameter, we swept over a range of values while keeping all other settings fixed. The results of the sensitivity analysis are summarized in Fig. 2. Overall, we observe that our method is reasonably robust. For grid size and sinusoidal period, performance remains stable across the tested ranges. For the noise level, small to moderate noise leads to similar performance, with a degradation only when the noise becomes large enough that it effectively corrupts the underlying structure of the data. For the hidden dimension, small dimensions impact the performance, but performance stabilizes once we enter a standard regime of model capacity.

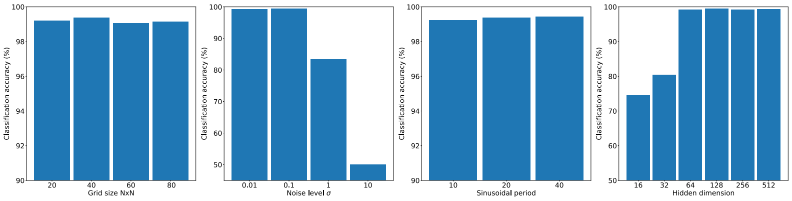

Figure 2: Hyperparameter sensitivity with respect to grid size, noise level, and hidden dimension.

G.6 Truncation Canonicalization with a pretrained classifier

We introduce in App. E.5 an application of our adaptive canonicalization on truncation prior maximization. We now illustrate the applicability of this setup with a pretrained image classifier. Specifically, we take a ResNet-18 (He et al., 2016) pretrained on ImageNet (Deng et al., 2009). We freeze the backbone, and train only the classifier using the CIFAR-10 or CIFAR-100 (Krizhevsky et al., 2009) training set. The experiment is conducted with ten independent runs, and the resulting image classification performance is reported in the Tab. 11. We see that truncation-based prior maximization improves classification performance over the standard vanilla baseline. This implies that our method enables the model to adaptively select a canonical truncation that enhances downstream performance. In addition, we observe that the selected canonical crops tend to tightly focus on the main object while discarding background and irrelevant context. It matches the intuition behind our prior maximization: by optimizing over the truncation family, the model chooses a representative transformed image that best aligns with its prior over the class. This experiment demonstrates that our adaptive canonicalization framework can be instantiated with a truncation symmetry and benefit from off-the-shelf pretrained models. It also highlights the potential of transformation families as a practical way to improve pretrained models via adaptive canonicalization.

Table 11: Image classification accuracy on CIFAR-10 and CIFAR-100 using a ResNet18 pretrained on ImageNet, with and without truncation canonicalization.

	CIFAR-10	CIFAR-100
Vanilla	72.09$\pm$1.0	56.94$\pm$0.8
Truncation canonicalization	74.92$\pm$0.6	60.38$\pm$0.5

G.7 Computational requirement comparison

Tab. 12 the training time per epoch with the number of parameters. We see that A-NLSF uses a similar number of parameters as the other methods and fewer than the spectral method. Its computational requirements are comparable to the other methods and does not rely on a significantly larger training budget than the competing methods.

Table 12: Running time per epoch(s)/number of parameters.

	MUTAG	PTC	ENZYMES	PROTEINS	NCI1
MLP	0.07/105K	0.10/114K	0.13/125K	0.37/129K	1.01/134K
GCN	0.40/116K	0.66/120K	0.81/137K	1.92/142K	5.84/149K
GAT	0.62/138K	0.87/149K	0.96/154K	2.34/159K	4.93/167K
GIN	0.14/105K	0.37/106K	0.52/107K	0.94/106K	1.97/121K
ChebNet	0.79/185K	1.25/189K	1.72/191K	3.64/217K	11.52/245K
FA+GIN	0.57/120K	1.04/123K	1.35/126K	2.55/130K	4.31/142K
OAP+GIN	0.22/105K	0.39/104K	0.57/109K	1.21/110K	2.36/124K
A-NLSF	0.44/132K	0.89/140K	1.29/145K	2.20/148K	4.37/151K

G.8 Training stability

To quantify the training stability of our method, we track the canonical rotations of a subset of 1500 randomly chosen training examples in the point cloud classification experiment. At each epoch, we measure the mean geodesic distance on $\mathcal{SO}(3)$ between the canonicalizations between consecutive epochs. Fig. 3 reports the mean geodesic distance between epochs. We observe that this distance decreases during the training and then remains stable, indicating that the canonical representatives stabilize with no rapid switching.

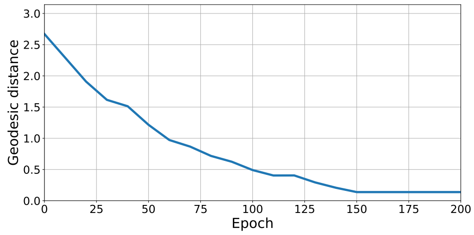

Figure 3: Mean geodesic distance on $\mathcal{SO}(3)$ between the canonicalizations between consecutive epochs.

G.9 Canonicalized point clouds

Fig. 4 shows the canonicalized point clouds for the chair class in the point cloud classification experiment. We randomly select 20 examples from this class and visualize them after applying the optimal transformations. We observe that the examples in this class share a similar orientation after canonicalization.

G.10 ShapeNet Part Segmentation

To expand our experimental study on point cloud data, we further conduct experiments on the ShapeNet part segmentation benchmark (Yi et al., 2016). The dataset consists of 16 shape categories annotated with a total of 50 parts, where each category is labeled with between two and six parts. Note that our prior maximization adaptive canonicalization method was naturally suited to classification tasks. Extending it to the segmentation task is not trivial, as the segmentation task requires predicting

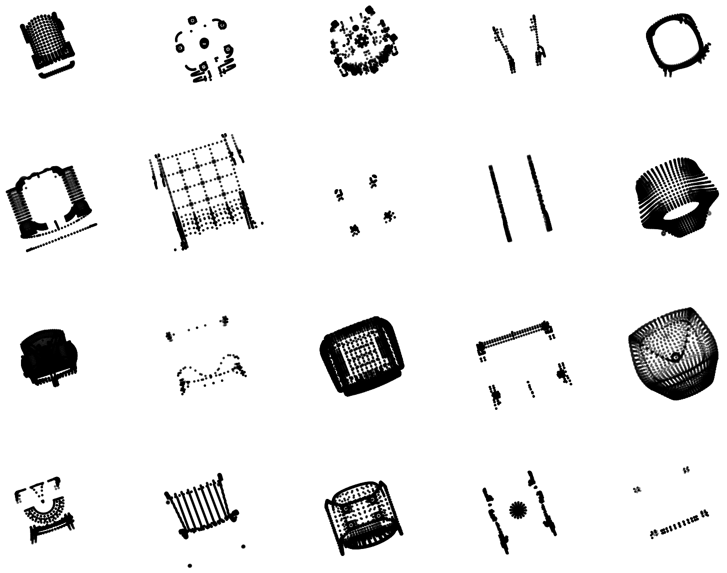

Figure 4: The canonicalized point clouds for the chair class.

a label for each point in the point cloud, and applying prior maximization to each point would be computationally inefficient. Therefore, in order to adapt our adaptive canonicalization to the segmentation task, we consider the adaptive canonicalization with the minimal entropy prior summing over nodes. This modification preserves the spirit of the adaptive canonicalization while making it compatible with per-point prediction. Tab. 13 reports the segmentation performance. For the PointNet backbone, we see that, similar to the point cloud classification task, the entropy-based adaptive canonicalization yields advantageous segmentation performance compared to equivariant architectures and standard canonicalization baselines. For the DGCNN backbone, our method attains performance comparable to equivariant architectures while outperforming existing canonicalization methods. These results demonstrate that our approach has potential beyond classification. Note that one of the main contributions of our work is to construct continuous and symmetry-respecting models. In the entropy prior adaptive canonicalization, the continuity property is not straightforward. We plan to investigate the continuity properties of this adapted approach in future work.

Table 13: Part segmentation performance on the ShapeNet part dataset. The metric is reported with the average category mean IoU Results of competing methods marked with * are taken from Deng et al. (2021); Kaba et al. (2023).

PointNet	62.3*
DGCNN	78.6*
VN-PointNet	72.8*
VN-DGCNN	81.4*
CN-PointNet	73.6$_{\pm 1.1}$ *
CN-DGCNN	78.5$_{\pm 0.9}$ *
AC-PointNet	76.0$_{\pm 0.6}$
AC-DGCNN	80.9$_{\pm 0.7}$

H Use of Large Language Models

Following the ICLR 2026 policy that requires disclosure of use of Large Language Models (LLMs), we state that an LLM was used for editing purposes, such as grammar, spelling, phrasing, and stylistic polish.