Proceedings Track

Equivariance with Learned Canonical Mappings

Editors: List of editors' names

Abstract

Symmetry-based neural networks often constrain the architecture in order to achieve invariance or equivariance to a group of transformations. In this paper, we propose an alternative that avoids this architectural constraint by learning to produce canonical representation of the data. These canonical mappings can readily be plugged into non-equivariant backbone architectures. We offer explicit ways to implement them for many groups of interest. We show that this approach enjoys universality while providing interpretable insights. Our main hypothesis is that learning a neural network to perform the canonicalization will perform better than doing it using predefined heuristics. Our results show that learning the canonical mappings indeed leads to better results and that the approach achieves great performance in practice.

Keywords: deep learning, symmetry, equivariance, shape recognition, group theory, vision

1. Introduction

The problem of designing machine learning models that properly exploit the structure and symmetry of the data is becoming more important as the field is broadening its scope to more complex problems. In multiple applications, the transformations with respect to which we require a model to be *invariant* or *equivariant* are known and provide a strong inductive bias (Bronstein et al., 2021; Bogatskiy et al., 2022; van der Pol et al., 2020; Mondal et al., 2020; Celledoni et al., 2021).

As is often the case, taking a step back and drawing analogies with human cognition is fruitful here. Human pattern recognition handles some symmetries with relative ease. When data is transformed in a way that preserves its essential characteristics, we precisely know if and how we should adapt our response. One context in which this has been particularly well-studied in cognitive science is visual shape recognition. Experiments have shown that subjects can accurately make the difference between different orientations of an object and actual modifications to the structure of an object (Shepard and Metzler, 1971; Carpenter and Eisenberg, 1978).

There are multiple ways in which this could be achieved. According to Tarr and Pinker (1989), theories of invariant shape recognition broadly fall into three categories: *viewpoint-independent* models, in which object representations depend only on invariants features, *multiple-view* models in which an object is represented as a set of representations corresponding to different orientations, and *single-view-plus-transformation* models in which an object is converted to a canonical orientation by a transformation process.

Correspondingly, similar ideas have been explored in deep learning to achieve equivariance. Models that impose equivariance through constraints in the architecture (Shawe-Taylor, 1989; Cohen and Welling, 2016; Ravanbakhsh et al., 2017) or that only use invariants as inputs (Villar et al., 2021) can be seen as belonging to the viewpoint-independent type. The multiple-view approach includes models that symmetrize the input by averaging over all

© 2022 .

the transformations or a subset of them (Manay et al., 2006; Benton et al., 2020; Yarotsky, 2022; Puny et al., 2022). By contrast, the transformation approach has seen less interest, and to our knowledge has not yet been used to achieve exact equivariance. This is all the more surprising considering that evidence from cognitive science suggests that this approach is used in human visual cognition (Shepard and Metzler, 1971; Carpenter and Eisenberg, 1978; Hinton and Parsons, 1981). When presented with a rotated version of an original pattern, the time taken by humans to do the association is proportional to the angle of rotation, which is more consistent with the hypothesis that we perform a *mental rotation*.

Some works (Jaderberg et al., 2015; Qi et al., 2017) have proposed to learn transformations of input to facilitate processing in a downstream task, but these approaches are closer to regularizers and provide no guarantees. Another method is to use heuristics to standardize inputs (Yüceer and Oflazer, 1993; Lowe, 2004; Aslan et al., 2022), but this approach requires significant hand-engineering and is hardly generalizable.

Present work We introduce a systematic and general method for equivariant machine learning based on learning mappings to canonical samples. We hypothesize that among all valid canonical mappings, some will lead to better downstream performance than others. Rather than trying to hand-engineer these functions, we may as well let them be learned in an end-to-end fashion with a prediction neural network. Our method can readily be used as an independent module that can be plugged into existing architectures to make them equivariant to arbitrary transformation groups, discrete or continuous. In addition to this flexibility, the proposed method has many advantages compared to other approaches. It is simpler, results in more easily interpretable models, and can be used to obtain highly expressive models.

Our contributions are as follow:

- **Novel Framework:** We introduce a general framework for equivariance to arbitrary groups based on mappings to canonical samples. This framework can be plugged into any existing non-equivariant architecture.
- **Theoretical Guarantees:** We prove that in some settings, such models are universal approximators of equivariant functions.
- **Efficient Implementations:** We provide multiple variants of efficient implementations of this framework to specific domains.
- **Practical Performance:** We perform experiments that show that the proposed method achieves excellent results in the image domain. We also validate our hypothesis that learning the canonical mapping is a better strategy than fixing it.

2. Canonical Mappings

2.1. Problem Setting

We are interested in learning functions $\boldsymbol{\phi} : \mathcal{X} \to \mathcal{Y}$ with inputs $\mathbf{x} \in \mathcal{X}$ belonging to an arbitrary vector space and similarly for outputs $\mathbf{y} \in \mathcal{Y}$. We will consider a set of linear symmetry transformations $T \subset \mathrm{GL}(\mathcal{X})$, where $\mathrm{GL}(\mathcal{X})$ is the set of invertible matrices over

the vector space $\mathcal{X}$. This is described by a group representation $\rho : G \to T$, where G is an abstract group. Without loss of generality, we can assume that ρ is a group isomorphism. Therefore, the inverse $\rho^{-1} : T \to G$ is defined. We also restrict ourselves to bounded representations, for which the induced operator norm of representation matrices is bounded by a positive number c, e.g. $\|\rho(g)\| \leq c \; \forall g \in G$. A function $\boldsymbol{\phi}$ is G-equivariant if

$$\boldsymbol{\phi}(\rho(g)\mathbf{x}) = \rho'(g)\boldsymbol{\phi}(\mathbf{x}), \; \forall \; g, x \in G \times \mathcal{X}, \tag{1}$$

where the group action ρ on the input and the group action ρ' on the output will be clear from the context. In particular, when $\rho'(g) = I$, we will say that $\boldsymbol{\phi}$ is invariant. We will use $\rho(G)$ to denote the image of ρ.

We call $\rho(G)\mathbf{x} = \{\rho(g)\mathbf{x} \mid \forall \; g \in G\}$ the orbit of the element $\mathbf{x}$. It is the set of elements to which $\mathbf{x}$ can be mapped to by the group action. The set of orbits, denoted $\mathcal{X}/G$ forms a partition of the set $\mathcal{X}$.

2.2. General Formulation

The invariance requirement on a function $\boldsymbol{\phi}$ amounts to having all the members of a group orbit mapped to the same image by $\boldsymbol{\phi}$. It is thus possible to achieve invariance by appropriately mapping all elements to a canonical sample from their orbit before applying any function. For equivariance, elements can be mapped to a canonical sample and after a function is applied, transformed back according to their original position in the orbit. This can be formalized by writing the equivariant function $\boldsymbol{\phi}$ in *canonicalized form* as

$$\boldsymbol{\phi}(\mathbf{x}) = h'(\mathbf{x})\,\mathbf{f}\left(h(\mathbf{x})^{-1}\mathbf{x}\right) \tag{2}$$

where the function $\mathbf{f} : \mathcal{X} \to \mathcal{Y}$ is called the prediction function and the function $h : \mathcal{X} \to \rho(G)$ is called the canonical mapping. Here $h(\mathbf{x})^{-1}$ is the inverse of the representation matrix and $h'(\mathbf{x}) = \rho'\left(\rho^{-1}(h(\mathbf{x}))\right)$ is the counterpart of $h(\mathbf{x})$ on the output.

Equivariance in equation 2 is obtained for any prediction function if the canonical mapping is itself G-equivariant, $h(\rho(g)\mathbf{x}) = \rho(g)h(\mathbf{x}) \;\; \forall \; g, x \in G \times \mathcal{X}$.

It may seem like the problem of obtaining an equivariant function has merely been transferred in this formulation. This is however not the case: in equation 2, the equivariance and prediction components are effectively decoupled. The canonical mapping h can therefore be chosen as a simple and inexpressive equivariant function, while the heavy-lifting is done by the prediction function $\mathbf{f}$.

A more general condition can be formulated, such that the decoupling is partial.

Theorem 1 *For some subgroup $K \leq G$, if there exists a $k \in K$ such that*

$$h(\rho(g)\mathbf{x}) = \rho(g)h(\mathbf{x})\rho(k)^{-1} \;\; \forall \; g, \mathbf{x} \in G \times \mathcal{X} \tag{3}$$

and the prediction function $\mathbf{f}$ is K-equivariant, then $\boldsymbol{\phi}$ defined in equation 2 is G-equivariant. If K is a normal subgroup such that $G = J \ltimes K$, this can be realized with a canonical mapping that has image $\rho(J)$, and that is J-equivariant and K-invariant.

The proof follows in Appendix A. This is equivalent to saying that the canonical mapping should output a coset in G/K in an equivariant way, the applied transformation being chosen arbitrarily within the coset. When $K = \{e\}$, only the canonical mapping is constrained, which is the case described above. The other extreme, given by $K = G$, corresponds to constraining the prediction function as is usually done : these are respectively the *single-view-plus-transformation* and the *viewpoint-independent* implementations described in the introduction.

2.3. Universality Result

We now introduce a more formal result on the universality of equivariant functions obtained with canonical mappings. A parameterized function $\boldsymbol{\phi}$ is a universal approximator of G-equivariant functions if for any G-equivariant continuous function $\boldsymbol{\psi}$, any compact set $\mathcal{K} \subseteq \mathcal{X}$ and any $\epsilon > 0$, there exists a choice of parameters such that $\|\boldsymbol{\psi}(\mathbf{x}) - \boldsymbol{\phi}(\mathbf{x})\| < \epsilon \ \forall \ \mathbf{x} \in \mathcal{K}$. We make the additional assumption that the set $\mathcal{K}$ is closed under the group action.

Theorem 2 *A G-equivariant parameterized function $\boldsymbol{\phi}$ written in canonical form (equation 2) and satisfying equation 3 with $K \leq G$, is a universal approximator of G-equivariant functions if the prediction function is a universal approximator of K-equivariant functions.*

The proof follows in Appendix B. The following corollary is especially relevant.

Corollary 3 *A G-equivariant parameterized function $\boldsymbol{\phi}$ written in canonical form with a G-equivariant canonical mapping and a multilayer perceptron (MLP) as a prediction function is a universal approximator of G-equivariant functions.*

This result can significantly simplify the design of universal approximators of equivariant functions since a non-universal equivariant architecture for the canonical mappings can be combined with an MLP.

3. Model Design for Canonical Mappings

In the next section, we elaborate on how suitable canonical mappings can be obtained in different settings.

3.1. Direct Approach and Optimization Approach

The canonical mapping can be chosen as any existing equivariant neural network architecture with the output being a group element; we call this the *direct approach*. For permutation groups and Lie groups, an equivariant multilayer perceptron (Shawe-Taylor, 1989; Finzi et al., 2021) can be used. We provide examples of implementations in the next section.

We also introduce an alternative method that can prove especially useful when dealing with continuous groups. The canonical mapping can be defined as

$$h(\mathbf{x}) = \underset{\rho(g) \in \rho(G)}{\arg\min} \ s(\rho(g)\mathbf{x}) \tag{4}$$

where $s : \mathcal{X} \to \mathbb{R}$ is a neural network. Such a mapping is G-equivariant provided that s has a unique minimum. We call this the *optimization approach*. Intuitively, the function

Proceedings Track

s represents the unnormalized distance between the input and the canonical sample of the orbit and is therefore minimized when $\rho(g)$ is the transformation that takes to the canonical sample. For continuous groups, the condition of the uniqueness of the optimum can be satisfied with Input Convex Neural Networks (ICNN) (Amos et al., 2017). The minimization in equation 4 can then be performed using any standard convex optimization method. For practical application, we are typically interested in low-dimensional Lie groups, which makes second-order optimization methods usable. The gradient of the neural network with respect to the parameters can be obtained efficiently (e.g. without having to differentiate through all the optimization steps) with implicit differentiation (Blondel et al., 2022).

The optimization implementation presents a close analogy with the mental rotation phenomenon described in the introduction, as humans try to minimize the distance between their representation of an object and the canonical one. As such, it is expected that the optimization process will take more iterations when the input sample is farther away in orbit from the canonical sample. This is consistent with the experimental evidence for mental rotation (Shepard and Metzler, 1971; Carpenter and Eisenberg, 1978).

3.2. Applications

3.2.1. Euclidean Group

The Euclidean group $E(d)$ is used to capture symmetry to rotations, translations, and reflections. Domains in which this type of symmetry is especially relevant include point cloud modeling, image processing and applications in physics. Below we give the design principles to get an equivariant model for image and point cloud inputs.

Image Input. In order to build networks equivariant to $E(2)$, we need a canonical mapping that outputs an element of $E(2)$ given an image. This can be achieved by using a G-CNN (Cohen and Welling, 2016) and taking an argmax of the output feature map. While G-CNN only considers four rotations this can be easily extended to any arbitrary rotations. This approach can be further simplified if we are using a translation equivariant prediction network which is generally the case with most CNN-based architectures. As the translation group $T(2)$ is a normal subgroup of the Euclidean group $E(2)$, Theorem 1 indicates that we only require the canonical mapping to be equivariant to $O(2)$. This means we can eliminate the spatial dimension in the output feature map of the canonical mapping and only need to take an argmax along the point group dimension to identify the correct orientation of the image. However, there are two problems with this approach. First, extending G-CNNs to higher-order rotations will lead to more computation and artifacts due to the finer rotation of filters. Second, we can not backpropagate through the canonical mapping as the argmax operation is not differentiable.

We can avoid the first problem by using a shallower network with a larger filter size. We empirically show why this is a sound choice for canonical mapping in Section 4. Furthermore to solve the second problem we can use the straight-through estimator trick (Bengio et al., 2013). Appendix D contains a PyTorch code snippet to perform the canonical mapping of images in a differentiable way using a G-CNN. Similarly, the output of the canonical mapping can be used to invert the feature maps from the prediction network back to their original orientation in a differentiable way.

Since CNNs are universal approximators of $T(2)$-equivariant functions (Yarotsky, 2022), it follows from Theorem 2 that a CNN augmented with an $O(2)$ equivariant canonical mapping is a universal approximator of $E(2)$-equivariant functions.

Point Cloud Input. Elements of the Euclidean group can be written as $(\mathbf{O}, \mathbf{t})$, where $\mathbf{O} \in \mathbb{R}^{n\times n}$ is an orthogonal matrix and $\mathbf{t} \in \mathbb{R}^n$ is an arbitrary translation vector. The vector representation on $n+1$ dimensional vectors (defined by concatenating a constant 1 to the original vectors) is then defined in the following way

$$\rho(\mathbf{O}, \mathbf{t}) = \begin{pmatrix} \mathbf{O} & \mathbf{t} \\ \mathbf{t}^T & 1 \end{pmatrix} \tag{5}$$

We seek to define an $E(d)$-equivariant canonical mapping. This can be done by defining it as $h(\mathbf{x}) = \rho\big(h^O(\mathbf{x}), h^t(\mathbf{x})\big)$, where the functions $h^O : \mathcal{X} \to \mathbb{R}^{n\times n}$ and $h^t : \mathcal{X} \to \mathbb{R}^n$ output the point transformation and the translation respectively. By the equivariance condition,

$$\big(h^O(\rho(\mathbf{O}, \mathbf{t})\mathbf{x}), h^t(\rho(\mathbf{O}, \mathbf{t})\mathbf{x})\big) = \big(\mathbf{O}h^O(\mathbf{x}), \mathbf{O}h^t(\mathbf{x}) + \mathbf{t}\big) \tag{6}$$

which means that h^O must be $O(d)$-equivariant and translation invariant, and that h^t must be $E(d)$-equivariant. These constraints can be satisfied by using already existing equivariant architectures. Since most of the work will be done by a prediction function that can be very expressive, like Pointnet (Qi et al., 2017), a simple and efficient architecture can be used for the canonical mapping, for example, Vector Neurons (Deng et al., 2021) or $E(d)$-Equivariant Graph Neural Networks (Satorras et al., 2021). The output of h^O can be made an orthogonal matrix by having it output n vectors and orthonormalizing them with the Gram-Schmidt procedure, which is itself equivariant.

Using Deep Sets (Zaheer et al., 2017) as a backbone architecture would result in a universal approximator of $E(d)$ and permutation equivariant functions, following Theorem 2 and Theorem 1 of (Segol and Lipman, 2020).

3.2.2. Symmetric Group

The symmetric group S_n over a finite set of n elements contains all the permutations of that set. This group captures the inductive bias that input order should not matter. Domains for which S_n-equivariance is desirable include object modelling and detection, graph representation learning, and applications in language modeling.

S_n-equivariant canonical mappings can be obtained both with a direct approach. In particular, the methods introduced by Mena et al. (2018); Cuturi et al. (2019); Zhang et al. (2019) can be used to obtain the permutation of the input. However, these approaches only learn relaxations of permutations and not actual elements of the permutation group. In addition, they offer no clear way to handle sets of different sizes.

We propose an alternative of S_n-equivariance for the canonical mapping that circumvents these issues. Instead of producing a permutation of the input, the canonical mapping can instead be designed to obtain a clustering of the input, with a function $c : \mathcal{X} \to \{1, \dots, k\}$, where k is the number of possible clusters. This can be made differentiable using the straight-through estimator described for the images above. Then, a corresponding permutation is obtained by sorting elements with respect to their cluster index, with elements

belonging to the same cluster being sorted arbitrarily. The canonical mapping would therefore respect equation 3 with respect to the subgroup $K = S_{l_1} \times \cdots \times S_{l_k}$, where l_i is the number of elements assigned to cluster i, the product of symmetric groups within clusters. Heuristically, this enforces the inductive bias that set elements should be mapped to high-level classes, across which interactions will be evaluated in an expressive way. This is close in spirit to what is done in the ClusterFormer (Wang et al., 2022) and Reformer (Kitaev et al., 2020) transformer architectures.

The prediction function $\mathbf{f}$ would then have to be $S_{l_1} \times \cdots \times S_{l_k}$ equivariant to obtain permutation equivariance. Such a function can readily be obtained with an equivariant multilayer perceptron, as derived by Wang et al. (2020).

4. Experiments

We perform a preliminary empirical analysis of the proposed framework in the image domain. Experiments on other domains will be the subject of more extensive work. We selected the Rotated MNIST dataset (Larochelle et al., 2007) which is used as a benchmark dataset to test equivariance in prior work (Cohen and Welling, 2016). We first compare our method with different CNN and G-CNN baselines. For CNN (base), we choose an architecture with 7 layers where layer 1 to 3 has 32, 4 to 6 has 64 and layer 7 has 128 channels respectively. Instead of pooling, we use convolution filters of size 5×5 with a stride 2 at layers 4 and 7. The remaining convolutions have filters of size 3×3 and stride 1. All the layers are followed by batch-norm and ReLU activation with dropout(p=0.4) only at layers 4 and 7. For the canonical mapping, we choose a shallow G-CNN with three layers. We start with a lifting layer with a filter that is the same size as the input image. This is followed by ReLU and group equivariant layers with 1×1 filters. We consider two variants: one with frozen weights for canonical mapping and the other where we learn it. For a pure G-CNN based baseline, we provide the value reported by Cohen and Welling (2016) and design a variant which has similar architecture to CNN (base) while matching the number of parameters of our model with canonical mapping. We call this G-CNN (p4 & =params). We consider two more variants of the CNN: one with the same number of parameters as our model and the other where the canonical mapping is done by finding the orientation of the digits using Principal Component Analysis (PCA). We train all the models for 100 epochs using Adam (Kingma and Ba, 2014) with a learning rate of 0.001. We perform early stopping based on the performance of the validation dataset with patience of 20 epochs.

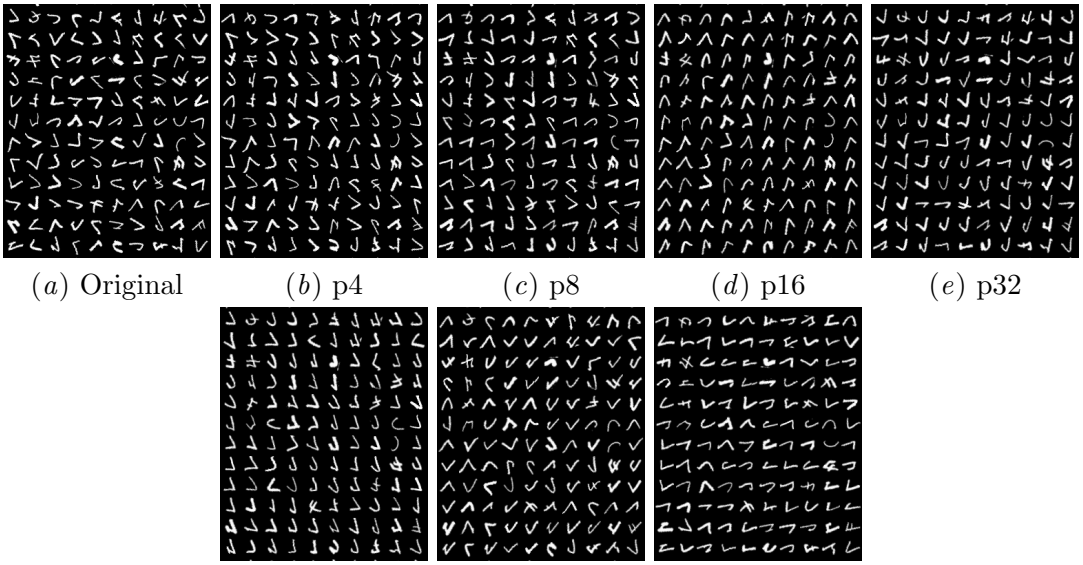
Method	**Error % ↓**
CNN (base)	4.9 ±0.2
G-CNN (p4)	2.28 ±0.004
G-CNN (p4 & = params)	2.36 ±0.15
CNN (= params)	4.8 ±0.37
CNN (w PCA)	3.35 ±0.21
Ours (p4 & frozen)	3.91 ±0.12
Ours (p4)	2.41 ±0.1

Table 1: Comparison with the existing work for Rotated-MNIST.

We see that using a fixed canonical mapping technique like PCA or canonical mapping with frozen parameters leads to an improvement in performance over the CNN baseline. However, learning canonical mapping provides a significant performance improvement. Our approach outperforms all the CNN-based baselines and is comparable to G-CNNs. We further notice that adding canonical mapping for reflections hurts performance. This is in

fact not surprising as reflection augmentations were not used to build the rotated-MNIST dataset. Adding reflection invariance while the data does not possess it may introduce ambiguity between certain digits like 2 and 5. Next, we perform experiments to understand the role of different components in our model using the group of n discrete rotations (pn).

4.1. Ablation Study



(a) Original (b) p4 (c) p8 (d) p16 (e) p32

(f) p64 (g) p64 (frozen) (h) PCA

Figure 1: Canonicalized images obtained from different canonical mappings for digit 7.

Num layers	**Order of the discrete rotation group**				
	p4	p8	p16	p32	p64
1	2.52 ±0.12	2.37 ±0.09	2.2 ±0.08	2.05 ±0.15	2.01 ±0.09
2	2.44 ±0.06	2.31 ±0.05	2.16 ±0.09	2.0 ±0.07	2.02 ±0.12
3	2.41 ±0.11	2.28 ±0.09	2.11 ±0.06	1.98 ±0.09	1.99 ±0.10

Table 2: Impact of the number of layers in canonical mapping network and order of the discrete rotations to which it is equivariant on the performance.

First, we vary both the number of layers of the canonical mapping network and the number of rotations it is equivariant to. For this, we extend the layers of GCNN to any arbitrary rotations. As we noticed that using a larger filter leads to better performance for higher order rotations we stick to architecture with a lifting layer that has image-sized filters followed by 1×1 filters. From Table 4.1, we notice that adding equivariance to higher order rotation in the canonical mapping leads to significant performance improvement in comparison to adding more layers. Figure 1 shows the canonical orientation resulting from the learnt canonical mapping with a single lifting layer on 110 randomly sampled images of class 7 from the test dataset. This suggests that having a shallow network is

Proceedings Track

enough to learn the correct canonical mapping with a sufficiently high order of discrete rotations. For p64, we see that all the similar-looking samples are aligned in one particular orientation. In contrast to this, although techniques like PCA or freezing parameters of the canonical mapping finds the correct canonical mapping for simple digits like 1 (see Appendix 3 they struggle to find the correct mapping for more complicated digits like 7. Next, we compare the inference time of our model with pure G-CNN-based architectures. For this experiment, we take the CNN architecture of our predictor network and replace the convolutions with group convolutions. As increasing the order of rotation in G-CNN requires more copies of rotated filters in the lifting layer and more parameters in the subsequent group convolution layers, we decreased the number of channels to keep the number of parameters the same as our model. Figure 2 shows that although G-CNN is slightly better for the p4 group increasing the order of discrete rotations improves our model's performance significantly compared to G-CNN. In addition to performance gain, our model's inference speed remains more or less constant while encoding invariance to higher-order rotations due to the shallow canonical mapping network. This makes our approach more suitable for building equivariance to bigger groups.

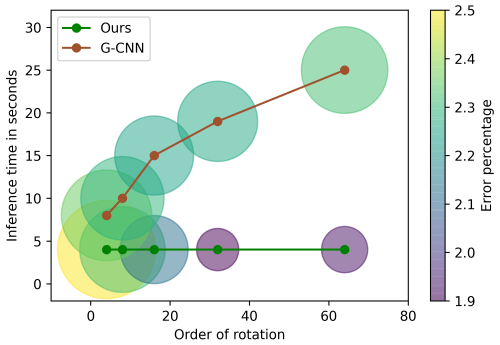


Figure 2: Inference time comparison of our method with G-CNN with increasing order of rotations.

5. Conclusion

In this work, we have proposed to use learned canonical mapping to obtain equivariant machine learning models. These canonical mappings can conveniently be plugged into existing architectures, resulting in highly expressive models. We have described general approaches to obtain canonical mappings and specific implementations for the Euclidean group (for images and point clouds) and for the symmetric group.

We performed preliminary experimental studies in the image domain to test our initial hypotheses regarding this framework. First, we show that our approach achieves comparable or better performance than baselines on invariant tasks. Importantly, learning the canonical network is a better approach than using a fixed mapping, either a frozen neural network or a heuristic approach like PCA. Our results also show that the canonical mapping can be realized with a shallow equivariant network, without hindering performance. Finally, we show that this approach reduces inference time and is more suitable for bigger groups compared to G-CNNs.

Areas of future work include an experimental study of this framework on other domains, such as point clouds and graphs. We also want to implement the optimization approach for learning the canonical mappings and compare it with the direct approach. Understanding how design choices for canonical mappings affect downstream performance would also be a potentially fruitful research direction.

References

Brandon Amos, Lei Xu, and J. Zico Kolter. Input convex neural networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 146–155. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/amos17b.html.

Benjamin Aslan, Daniel Platt, and David Sheard. Group invariant machine learning by fundamental domain projections, 2022. URL https://arxiv.org/abs/2202.02164.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Gregory Benton, Marc Finzi, Pavel Izmailov, and Andrew G Wilson. Learning invariances in neural networks from training data. *Advances in neural information processing systems*, 33:17605–17616, 2020.

Mathieu Blondel, Quentin Berthet, marco cuturi, Roy Frostig, Stephan Hoyer, Felipe Llinares-López, Fabian Pedregosa, and Jean-Philippe Vert. Efficient and modular implicit differentiation, 2022. URL https://openreview.net/forum?id=TQ75Md-FqQp.

Alexander Bogatskiy, Sanmay Ganguly, Thomas Kipf, Risi Kondor, David W Miller, Daniel Murnane, Jan T Offermann, Mariel Pettee, Phiala Shanahan, Chase Shimmin, et al. Symmetry group equivariant architectures for physics. *arXiv preprint arXiv:2203.06153*, 2022.

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

Patricia A. Carpenter and Peter Eisenberg. Mental rotation and the frame of reference in blind and sighted individuals. *Perception & Psychophysics*, 23(2):117–124, 1978. doi: 10.3758/BF03208291. URL https://doi.org/10.3758/BF03208291.

Elena Celledoni, Matthias J Ehrhardt, Christian Etmann, Brynjulf Owren, Carola-Bibiane Schönlieb, and Ferdia Sherry. Equivariant neural networks for inverse problems. *Inverse Problems*, 37(8):085006, 2021.

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

Marco Cuturi, Olivier Teboul, and Jean-Philippe Vert. Differentiable ranking and sorting using optimal transport. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/d8c24ca8f23c562a5600876ca2a550ce-Paper.pdf.

Proceedings Track

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*, pages 12200–12209, 2021.

Marc Finzi, Max Welling, and Andrew Gordon Wilson. A practical method for constructing equivariant multilayer perceptrons for arbitrary matrix groups. *arXiv preprint arXiv:2104.09459*, 2021.

Geoffrey E Hinton and Lawrence M Parsons. Frames of reference and mental imagery. *Attention and performance IX*, pages 261–277, 1981.

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

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

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rkgNKkHtvB.

Hugo Larochelle, Dumitru Erhan, Aaron Courville, James Bergstra, and Yoshua Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, page 473–480, New York, NY, USA, 2007. Association for Computing Machinery. ISBN 9781595937933. doi: 10.1145/1273496.1273556. URL https://doi.org/10.1145/1273496.1273556.

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

Siddharth Manay, Daniel Cremers, Byung-Woo Hong, Anthony J Yezzi, and Stefano Soatto. Integral invariants for shape matching. *IEEE Transactions on pattern analysis and machine intelligence*, 28(10):1602–1618, 2006.

Gonzalo Mena, David Belanger, Scott Linderman, and Jasper Snoek. Learning latent permutations with gumbel-sinkhorn networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Byt3oJ-0W.

Arnab Kumar Mondal, Pratheeksha Nair, and Kaleem Siddiqi. Group equivariant deep reinforcement learning. *arXiv preprint arXiv:2007.03437*, 2020.

Omri Puny, Matan Atzmon, Edward J. Smith, Ishan Misra, Aditya Grover, Heli Ben-Hamu, and Yaron Lipman. Frame averaging for invariant and equivariant network design. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=zIUyj55nXR.

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*, pages 652–660, 2017.

Siamak Ravanbakhsh, Jeff Schneider, and Barnabas Poczos. Equivariance through parameter-sharing. In *International Conference on Machine Learning*, pages 2892–2901. PMLR, 2017.

Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E (n) equivariant graph neural networks. *arXiv preprint arXiv:2102.09844*, 2021.

Nimrod Segol and Yaron Lipman. On universal equivariant set networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HkxTwkrKDB.

J. Shawe-Taylor. Building symmetries into feedforward networks. In *1989 First IEE International Conference on Artificial Neural Networks, (Conf. Publ. No. 313)*, pages 158–162, 1989.

N. Shepard and Jacqueline Metzler. Mental rotation of three-dimensional objects. *Science*, pages 701–703, 1971.

Michael J Tarr and Steven Pinker. Mental rotation and orientation-dependence in shape recognition. *Cognitive Psychology*, 21(2):233–282, 1989. ISSN 0010-0285. doi: https://doi.org/10.1016/0010-0285(89)90009-1. URL https://www.sciencedirect.com/science/article/pii/0010028589900091.

Elise van der Pol, Daniel Worrall, Herke van Hoof, Frans Oliehoek, and Max Welling. Mdp homomorphic networks: Group symmetries in reinforcement learning. *Advances in Neural Information Processing Systems*, 33:4199–4210, 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. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=ba27-RzNaIv.

Ningning Wang, Guobing Gan, Peng Zhang, Shuai Zhang, Junqiu Wei, Qun Liu, and Xin Jiang. ClusterFormer: Neural clustering attention for efficient and effective transformer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2390–2402, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.170. URL https://aclanthology.org/2022.acl-long.170.

Renhao Wang, Marjan Albooyeh, and Siamak Ravanbakhsh. Equivariant maps for hierarchical structures, 2020.

Dmitry Yarotsky. Universal approximations of invariant maps by neural networks. *Constructive Approximation*, 55(1):407–474, 2022.

Cem Yüceer and Kemal Oflazer. A rotation, scaling, and translation invariant pattern classification system. *Pattern recognition*, 26(5):687–710, 1993.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3391–3401. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/6931-deep-sets.pdf.

Yan Zhang, Jonathon Hare, and Adam Prügel-Bennett. Learning representations of sets through optimized permutations. In *International Conference on Learning Representations*, 2019.

Appendix A. Proof of Theorem 1

We first show equivariance for a general subgroup K. We start with

$$\boldsymbol{\phi}\left(\rho\left(g\right)\mathbf{x}\right) = h'\left(\rho\left(g\right)\mathbf{x}\right)\mathbf{f}\left(h\left(\rho\left(g\right)\mathbf{x}\right)^{-1}\rho\left(g\right)\mathbf{x}\right) \tag{7}$$

If equation 3 is satisfied, then there is a $k \in K$ such that

$$\boldsymbol{\phi}\left(\rho\left(g\right)\mathbf{x}\right) = \rho'\left(g\right)h'\left(\mathbf{x}\right)\rho'\left(k\right)^{-1}\mathbf{f}\left(\left[\rho\left(g\right)h\left(\mathbf{x}\right)\rho\left(k\right)^{-1}\right]^{-1}\rho\left(g\right)\mathbf{x}\right) \tag{8}$$

$$\boldsymbol{\phi}\left(\rho\left(g\right)\mathbf{x}\right) = \rho'\left(g\right)h'\left(\mathbf{x}\right)\rho'\left(k\right)^{-1}\mathbf{f}\left(\rho\left(k\right)h\left(\mathbf{x}\right)^{-1}\rho\left(g\right)^{-1}\rho\left(g\right)\mathbf{x}\right) \tag{9}$$

Using the K-equivariance of $\mathbf{f}$, we obtain

$$\boldsymbol{\phi}\left(\rho\left(g\right)\mathbf{x}\right) = \rho'\left(g\right)h'\left(\mathbf{x}\right)\rho'\left(k\right)^{-1}\rho'\left(k\right)\mathbf{f}\left(h\left(\mathbf{x}\right)^{-1}\mathbf{x}\right) \tag{10}$$

$$\boldsymbol{\phi}\left(\rho\left(g\right)\mathbf{x}\right) = \rho'\left(g\right)h'\left(\mathbf{x}\right)\mathbf{f}\left(h\left(\mathbf{x}\right)^{-1}\mathbf{x}\right) \tag{11}$$

When K is a normal subgroup of G, the group can be taken to be isomorphic to a semidirect product $G \simeq K \rtimes J$. Group elements can be written as $g = (k, j)$, where $k \in K$ and $j \in J$. The product is defined as $g_1 g_2 = (k_1, j_1)(k_2, j_2) = (k_1 \varphi[j_1](k_2), j_1 j_2)$, where $\varphi : J \to \mathrm{Aut}(K)$ is a group homomorphism. Setting $k_2 = e$ and $j_1 = e$, we get any group element as $(k_1, e)(e, j_2) = (k_1, j_2)$.

If the canonical mapping is J-equivariant and K-invariant, we have

$$h\left(\rho\left(k,j\right)\mathbf{x}\right) = h\left(\rho\left(k,e\right)\rho\left(e,j\right)\mathbf{x}\right) \tag{12}$$

$$h\left(\rho\left(k,j\right)\mathbf{x}\right) = \rho\left(e,j\right)h\left(\mathbf{x}\right) \tag{13}$$

We then show that there is a $k' \in K$ such that equation 3 is satisfied

$$\rho\left(e,j\right)h\left(\mathbf{x}\right) = \rho\left(k,e\right)\rho\left(e,j\right)h\left(\mathbf{x}\right)\rho\left(k',e\right)^{-1} \tag{14}$$

Using the fact that conjugation of an element of K by an element of G preserves K membership, we obtain

$$\rho(e,j)\,h(\mathbf{x}) = \rho(k,e)\,\rho(e,j)\,h(\mathbf{x})\,h(\mathbf{x})^{-1}\,\rho\left(k'',e\right)^{-1}h(\mathbf{x}) \tag{15}$$

$$\rho(e,j)\,h(\mathbf{x}) = \rho(k,e)\,\rho(e,j)\,\rho\left(k'',e\right)^{-1}h(\mathbf{x}) \tag{16}$$

$$\rho(e,j)\,h(\mathbf{x}) = \rho(k,e)\,\rho(e,j)\,\rho(e,j)^{-1}\,\rho\left(k''',e\right)^{-1}\rho(e,j)\,h(\mathbf{x}) \tag{17}$$

$$\rho(e,j)\,h(\mathbf{x}) = \rho(k,e)\,\rho\left(k''',e\right)^{-1}\rho(e,j)\,h(\mathbf{x}) \tag{18}$$

equality is satisfied for $k = k'''$.

Finally, we show that in this case, the image of h is $\rho(J)$. To show this, we first remark that in each orbit $\mathcal{X}/G$ of the group action, the canonical sample $\hat{\mathbf{x}}$ can be obtained by from any orbit member $\mathbf{x}$, as $\hat{\mathbf{x}} = h(\mathbf{x})^{-1}\mathbf{x}$. For the canonical sample, we have $h(\hat{\mathbf{x}}) = \rho(e,e)$. Since any orbit member can conversely be written as $\mathbf{x} = \rho(k,j)\,\hat{\mathbf{x}}$ for some $k \in K$ and $j \in J$, we have

$$h(\mathbf{x}) = h(\rho(k,j)\,\hat{\mathbf{x}}) \tag{19}$$

$$h(\mathbf{x}) = \rho(e,j)\,h(\hat{\mathbf{x}}) \tag{20}$$

$$h(\mathbf{x}) = \rho(e,j) \tag{21}$$

which completes the proof.

Appendix B. Proof of Theorem 2

Proof The proof is inspired by the proof of Theorem 4 from (Puny et al., 2022).

Let $\boldsymbol{\psi}$ be an arbitrary G-equivariant function, and $\boldsymbol{\phi}$ be defined by equation 2. By the equivariance of $\boldsymbol{\psi}$, we have

$$\|\boldsymbol{\psi}(\mathbf{x}) - \boldsymbol{\phi}(\mathbf{x})\| = \left\|h(\mathbf{x})\,\boldsymbol{\psi}\left(h(\mathbf{x})^{-1}\mathbf{x}\right) - h(\mathbf{x})\,\mathbf{f}\left(h(\mathbf{x})^{-1}\mathbf{x}\right)\right\| \tag{22}$$

By the boundedness of representation matrices, we then obtain

$$\|\boldsymbol{\psi}(\mathbf{x}) - \boldsymbol{\phi}(\mathbf{x})\| \leq \|h(\mathbf{x})\|\left\|\boldsymbol{\psi}\left(h(\mathbf{x})^{-1}\mathbf{x}\right) - \mathbf{f}\left(h(\mathbf{x})^{-1}\mathbf{x}\right)\right\| \tag{23}$$

$$\|\boldsymbol{\psi}(\mathbf{x}) - \boldsymbol{\phi}(\mathbf{x})\| \leq c\left\|\boldsymbol{\psi}\left(h(\mathbf{x})^{-1}\mathbf{x}\right) - \mathbf{f}\left(h(\mathbf{x})^{-1}\mathbf{x}\right)\right\| \tag{24}$$

where $c > 0$.

If $\mathbf{f}$ is a universal approximator of K-equivarant functions, then it is also a universal approximator of G-equivariant functions. We therefore have

$$\|\boldsymbol{\psi}(\mathbf{x}) - \mathbf{f}(\mathbf{x})\| \leq \epsilon,\ \forall\ \mathbf{x} \in \mathcal{K} \tag{25}$$

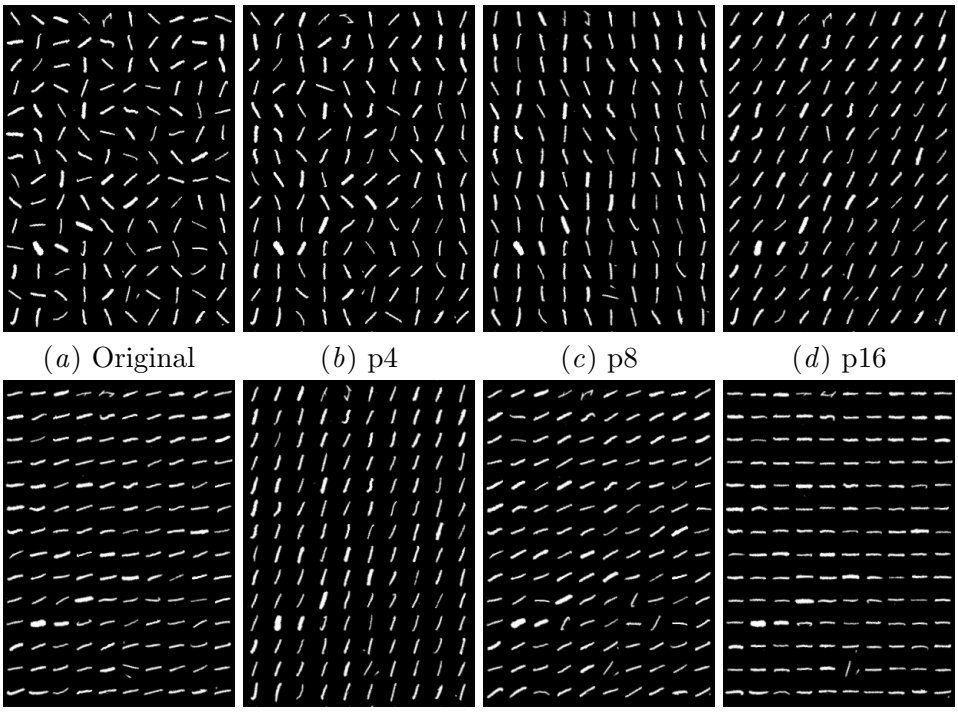
Using the fact that $\mathcal{K}$ is closed under the group action, we obtain the desired result

$$\|\boldsymbol{\psi}(\mathbf{x}) - \boldsymbol{\phi}(\mathbf{x})\| \leq \epsilon,\ \forall\ \mathbf{x} \in \mathcal{K} \tag{26}$$

■

Proceedings Track

Appendix C. Additional results



(*a*) Original (*b*) p4 (*c*) p8 (*d*) p16

(*e*) p32 (*f*) p64 (*g*) p64 (frozen) (*h*) PCA

Figure 3: Canonicalized images obtained from different canonical mappings for digit 1.

Appendix D. Algorithm for Image Inputs

Algorithm 1 Differentiable Canonicalization for Image Inputs

```
import torch.nn.functional as F
import kornia as K

def get_canonicalized_images(images, fibre_features, use_reflection=True):
    """
    images shape: (batch_size, in_channels, height, width)
    fibres_activations shape: (batch_size, num_group_elements)
    :return: (batch_size, in_channels, height, width)
    """
    num_group_elements = fibre_features.shape[-1]
    num_rotations = num_group_elements // 2 if use_reflection else num_group_elements

    fibre_features_one_hot = F.one_hot(
        torch.argmax(fibre_activations, dim=-1),
        num_group_elements
    ).float()

    fibre_features_soft = F.softmax(fibre_activations, dim=-1)
    ref_angles = torch.linspace(0., 360., num_rotations+1)[:num_rotations]

    if use_reflection:
        ref_angles = torch.cat([ref_angles, ref_angles], dim=0)

    angles = torch.sum((
        fibre_features_one_hot + fibre_features_soft - fibre_features_soft.detach()
        ) * ref_angles, dim=-1)

    if use_reflection:
        reflect_one_hot = torch.cat(
            [torch.zeros(num_rotations), torch.ones(num_rotations)]
            , dim=0)
        reflect_indicator = torch.sum((
            fibre_features_one_hot + fibre_features_soft - fibre_features_soft.detach()
            ) * reflect_one_hot, dim=-1)

        x_reflected = K.geometry.hflip(x)
        reflect_indicator = reflect_indicator[:,None,None,None]
        x = (1 - reflect_indicator) * x + reflect_indicator * x_reflected

    x = K.geometry.rotate(x, -angles)
    return x


# Use a shallow G-CNN as a canonicalization_network
feature_map = canonicalization_network(images)
# feature_map shape: (batch_size, num_channels, num_group_elements, height, width)

fibre_features = feature_map.mean(dim=(1, 3, 4))
# fibre_features shape: (batch_size, num_group_elements)

canonicalized_images = get_canonicalized_images(images, fibre_featuresr)
```