Group Equivariant Convolutional Networks

Tristan Lecourtois
ENS Paris-Saclay
tristan.lecourtois@ens-paris-saclay.fr

Léos Coutrot
ENS Paris-Saclay
leos.coutrot@ens-paris-saclay.fr

Abstract

Group-Equivariant Convolutional Networks (G-CNNs) are a class of neural network architectures that incorporate geometric symmetries, such as discrete rotation, translation, and reflection, by generalizing the weight-sharing principle of standard CNNs. This design avoids the need for a naive model to explicitly learn these symmetries through data augmentation, thereby reducing model complexity and accelerating training. This synthesis provides a brief summary of the G-CNN theory, illustrating its application in handling discrete group symmetries. It then details advanced theoretical approaches that extend the core principles of G-CNNs to achieve equivariance for continuous transformations.

Keywords

Equivariance, G-CNNs, Symmetry Groups, Steerable Kernels, discrete groups, Harmonic Networks

ACM Reference Format:
Tristan Lecourtois and Léos Coutrot. 2026. Group Equivariant Convolutional Networks. In . ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Deep learning has become essential for analyzing structured data in fields such as computer vision and molecular modeling. In many of these applications, inputs possess natural geometric symmetries: images can be translated or rotated, molecules can be reoriented, and sets can be permuted without changing their meaning. Mathematically, these transformations are captured by symmetry groups, and respecting them is crucial for producing physically and semantically consistent predictions. Depending on the task, the model may need to be invariant (the prediction of a molecule's atomization energy doesn't depend on the molecule's rotation) or equivariant (predicted forces must rotate with the molecule). In principle, generic neural networks are capable of acquiring the appropriate invariance or equivariance properties from data, but a model would have to learn this explicitly. It would need to be shown samples in every possible transformed pose before it would understand their equivariance, resulting in increased sample complexity, extended training times, and reduced robustness to unseen transformations.

An alternative is to impose the relevant symmetry constraints directly at the architectural level. By constructing models that are equivariant by design, one ensures that learned representations are consistent under the prescribed group actions. This equivariance by design reduces the model's complexity and the number of parameters, which frees learning capacity, accelerates the training process, and leads to better performance.

Convolutional Neural Networks (CNNs) achieve this for translations: their sparse filters and shared parameters make them translation-equivariant, enabling reliable detection of local patterns regardless of spatial position (a formal proof is provided in Appendix A). However, standard CNNs do not automatically generalize to other symmetries such as rotations or reflections, which are fundamental in many perception tasks.

Cohen and Welling [1] introduce G-CNNs, a generalization of standard CNNs to arbitrary symmetry groups. Unlike traditional CNNs, which are constrained to translation equivariance, G-CNNs extend the convolutional operation to functions defined on groups, ensuring that feature maps transform consistently under any group action. This framework allows the network to exploit richer symmetries inherent in the data, leading to improved sample efficiency and stronger generalization. In the following sections, we formalize the notion of group convolution and describe how G-CNNs can be systematically constructed for both discrete and continuous groups. We applied algorithms from [1] and reproduced results on the rotated MNIST dataset.

2 Background

2.1 Groups and Equivariance

A *group* G is a set equipped with a binary operation $* : G \times G \to G$ that satisfies four fundamental properties. First, the group is *closed* under the operation: for all $g, h \in G$, the result $g * h$ is also in G. Second, the operation is *associative*, meaning $(g*h)*k = g*(h*k)$ for all $g, h, k \in G$. Third, there exists an *identity element* $e \in G$ such that $e * g = g * e = g$ for all $g \in G$. Finally, every element $g \in G$ has an *inverse* $g^{-1} \in G$ satisfying $g * g^{-1} = g^{-1} * g = e$.

Let X be a space on which G acts. A function $f : X \to \mathbb{R}^K$ is said to be *equivariant* to the group G if applying a group transformation $g \in G$ to the input results in a predictable transformation of the output, i.e.,

$$f(L_g x) = L'_g f(x), \quad \forall g \in G,\, x \in X,$$

where L_g and L'_g denote the actions of G on the input and output spaces, respectively. If L'_g is the identity for all g, the function is *invariant* under the group action.

2.2 Translation and Rotation Groups in Images

In the context of 2D images, the most commonly considered symmetry groups are discrete translations, rotations, and their combinations. The group of planar translations and rotations by multiples of 90° is denoted as $p4$. Formally, it can be written as a semidirect

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

product:

$$p4 = \mathbb{Z}^2 \rtimes C_4,$$

where C_4 is the cyclic group of order 4 representing rotations by $0°, 90°, 180°$ and $270°$. The semidirect product $\rtimes$ indicates that rotations act non-trivially on translations and the effect of a translation depends on the rotation applied.

The group $p4m$ extends $p4$ by including reflections along horizontal, vertical, and diagonal axes. This forms a discrete group capturing both rotational and mirror symmetries. Like $p4$, $p4m$ can be represented in terms of the combination of translations, rotations, and reflection operations. Recall that these groups all act on pixels, not images.

To apply these groups to images, we interpret an image (or a feature map) as a function over pixel coordinates, assigning a value to each pixel. While a group element $g \in G$ is formally defined to act on pixel locations, we need a corresponding action on the entire image. This is achieved by defining

$$L_g(f) = f \circ g^{-1},$$

which maps the input image $f : \mathbb{Z}^2 \to \mathbb{R}$ to its transformed version $L_g(f)$. The inverse g^{-1} ensures that translating or rotating the image corresponds to moving pixels in the opposite direction, keeping the resulting image consistent with the intended transformation.

This construction generalizes to any discrete symmetry group, such as $p4$ or $p4m$, and forms the foundation for designing convolutional layers that are equivariant by construction. It guarantees that the feature maps respond predictably under the action of group transformations, which is essential for lifting and group convolutions in G-CNNs.

2.3 Equivariance in $p4$ and $p4m$

When a convolutional network is equivariant to a group such as $p4$, its feature maps transform consistently under any translation or $90°$ rotation. Equivariance to $p4m$ further guarantees consistent behavior under reflections. Concretely, for a feature map $f : \mathbb{Z}^2 \to \mathbb{R}^K$ and a group element $g \in p4$ or $g \in p4m$, we have

$$(f \star \psi)(gx) = g(f \star \psi)(x),$$

where $\star$ denotes the group convolution operation. This property ensures that patterns detected by the network are recognized regardless of the discrete symmetries of the input.

3 Group Convolutions

3.1 Lifting Convolution

The first layer of a G-CNN, called the *lifting convolution*, maps a signal defined on $\mathbb{Z}^2$ to a signal on the group G:

$$f_{in} : \mathbb{Z}^2 \to \mathbb{R} \quad \longrightarrow \quad f_{out} : G \to \mathbb{R}, \quad f_{out} = f_{in} * \psi.$$

Formally, the lifting convolution is defined as

$$[f * \psi](g) = \sum_{\mathbf{y} \in \mathbb{Z}^2} f_{in}(\mathbf{y})\, \psi(g^{-1}\mathbf{y}),$$

where $g \in G$.

For the case $G = \mathbb{Z}^2 \rtimes C_4$, each group element can be parametrized by a translation $\mathbf{x} \in \mathbb{Z}^2$ and a rotation $r \in C_4$, such that $g = (\mathbf{x}, r)$.

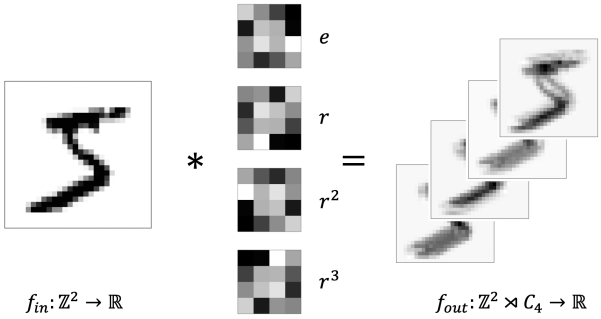

Figure 1: Illustration of lifting convolution. The input image is convolved with a single filter applied in four rotations, producing four output feature maps corresponding to each rotation.

Then the lifting convolution can be written as

$$\begin{aligned} f_{out}(\mathbf{x}, r) &= (f_{in} * \psi)(\mathbf{x}, r) \\ &= \sum_{\mathbf{y} \in \mathbb{Z}^2} f_{in}(\mathbf{y})\, L_r L_{\mathbf{x}} \psi(\mathbf{y}) \\ &= \sum_{\mathbf{y} \in \mathbb{Z}^2} f_{in}(\mathbf{y})\, L_r \psi(\mathbf{y} - \mathbf{x}) \\ &= \sum_{\mathbf{y} \in \mathbb{Z}^2} f_{in}(\mathbf{y})\, \psi_r(\mathbf{y} - \mathbf{x}), \end{aligned}$$

where $L_{\mathbf{x}}$ and L_r denote the translation and rotation operators applied to the filter, and ψ_r is the rotated version of the filter for each $r \in C_4$.

Intuitively, if $f_{in} : \mathbb{Z}^2 \to \mathbb{R}$ is the input image and $\{\psi_r\}_{r \in C_4}$ is a bank of four filters corresponding to rotations e, r, r^2, r^3, then the output $f_{out} : \mathbb{Z}^2 \times C_4 \to \mathbb{R}$ encodes responses at every spatial location and rotation.

3.2 Group Convolution

For subsequent layers, both the input and filters are functions on the group G, i.e., $f, \psi : G \to \mathbb{R}$. The group convolution is defined as

$$[f * \psi](g) = \sum_{h \in G} f(h)\, \psi(g^{-1}h), \quad g \in G.$$

For $G = \mathbb{Z}^2 \rtimes C_4$, this expands to

$$\begin{aligned} f_{out}(\mathbf{x}, r) &= (f_{in} * \psi)(\mathbf{x}, r) \\ &= \sum_{\mathbf{y} \in \mathbb{Z}^2} \sum_{r' \in C_4} f_{in}(\mathbf{y}, r')\, L_r L_{\mathbf{x}} \psi(\mathbf{y}, r') \\ &= \sum_{\mathbf{y} \in \mathbb{Z}^2} \sum_{r' \in C_4} f_{in}(\mathbf{y}, r')\, L_r \psi(\mathbf{y} - \mathbf{x}, r') \\ &= \sum_{\mathbf{y} \in \mathbb{Z}^2} \sum_{r' \in C_4} f_{in}(\mathbf{y}, r')\, \psi_r(\mathbf{y} - \mathbf{x}, r') \\ &= \sum_{\mathbf{y} \in \mathbb{Z}^2} \sum_{r' \in C_4} f_{in}(\mathbf{y}, r')\, \psi(r^{-1}(\mathbf{y} - \mathbf{x}), r^{-1}r'). \end{aligned}$$

Here, the input feature map f_{in} has an additional group dimension $r' \in C_4$, in addition to the usual spatial dimensions over $\mathbb{Z}^2$.

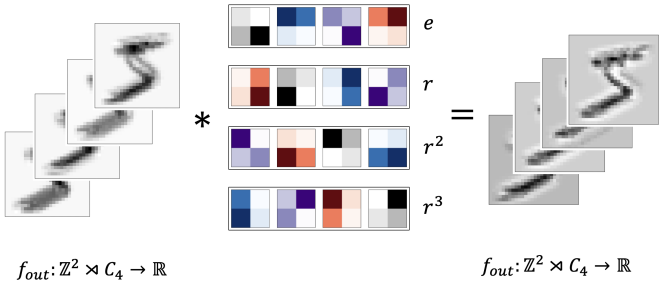

Figure 2: Illustration of group convolution. The input feature maps from the previous layer, which already have 4 orientations per spatial location, are convolved with a bank of 4 filters, each applied in 4 rotations, resulting in 16 output feature maps.

Each layer preserves equivariance to the group G, allowing stacking multiple group convolutions while maintaining structured symmetry in the representations.

3.3 Pointwise Nonlinearities and Pooling

After defining G-convolutions, we also need to ensure that other operations in CNNs, such as nonlinearities and pooling, preserve equivariance. Feature maps in G-CNNs are functions on a group G. Applying a standard nonlinearity (e.g., ReLU) elementwise to each feature vector preserves equivariance, since the transformation acts independently on each group element:

$$(C_v f)(g) = v(f(g)), \quad g \in G.$$

Equivariance is preserved because no mixing occurs between channels corresponding to different group elements. Pooling can also be made equivariant by pooling over neighborhoods defined as left-translated regions in the group. Strided pooling corresponds to subsampling on a subgroup $H \subset G$, which preserves equivariance with respect to H. Choosing pooling regions as cosets of a subgroup allows reducing feature map size while maintaining equivariance in the quotient space G/H.

4 Experiments

To evaluate the effectiveness of equivariant CNNs, we conducted experiments on four datasets commonly used in image classification and equivariance studies: MNIST, Rotated MNIST, CIFAR-10, and augmented CIFAR-10. MNIST consists of grayscale images of handwritten digits, while Rotated MNIST applies random rotations to each digit, testing the model's robustness to rotational transformations. CIFAR-10 contains natural RGB images from 10 classes, and augmented CIFAR-10 includes standard data augmentation such as random rotations, flips, and translations to increase variability.

Our first baseline model consists of 5 convolutional layers with 3×3 kernels, with 8, 16, 32, 64, and 128 channels respectively. Each layer uses ReLU activations, batch normalization, dropout, and 3D max-pooling to improve generalization and reduce overfitting.

We then compared this baseline with equivariant CNNs designed to leverage rotational symmetries. Specifically, we tested p2, p4, p8, and p16 CNNs, corresponding to architectures equivariant to 2, 4, 8, and 16 discrete rotations respectively.

Table 1: Test accuracy (%) of baseline CNN and equivariant CNNs on different datasets.

Dataset	CNN	P2CNN	P4CNN	P8CNN	P16CNN
MNIST	98.5	98.7	99.0	99.2	**99.3**
Rotated MNIST	92.0	94.5	96.0	97.2	**97.5**
CIFAR-10	81.3	82.1	83.0	83.5	**84.0**
Augmented CIFAR-10	82.0	83.0	84.2	85.0	**85.5**

Table 2 shows that accuracy increases with higher equivariance for all datasets. On Rotated MNIST, the baseline CNN achieves 92.0% while P16CNN reaches 97.5%, showing improved generalization to unseen rotations. Even on MNIST and CIFAR-10 without extreme rotations, equivariant models outperform the baseline, highlighting parameter efficiency and better generalization. On augmented CIFAR-10, P16CNN achieves 85.5%, confirming effectiveness under spatial transformations.

For instance, the Figure 3 shows that the convergence rates and the final validation losses of CNNs improve with an increasing level of equivariance. The faster convergence of equivariant CNNs relates to their enhanced data efficiency: each training sample is automatically generalized to any P_n-transformed pose, where n denotes the number of discrete rotations, such that fewer samples need to be presented before the network learns a given task.

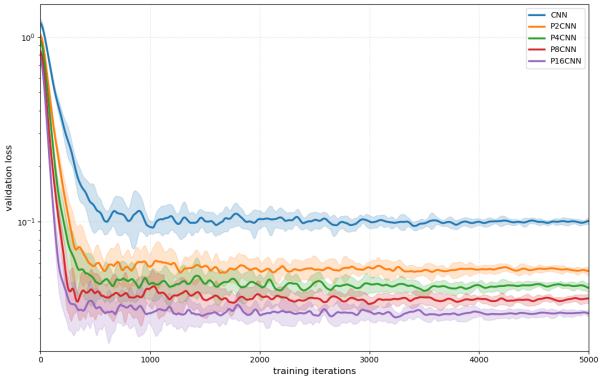

Figure 3: Validation Loss on Rotated MNIST for different models

However, regular P_n-equivariant CNNs remain limited to discrete rotational symmetries; they cannot guarantee equivariance to arbitrary continuous rotations. This limitation motivates the use of steerable architectures and Harmonic Networks, which achieve true continuous rotational equivariance.

5 Experiment : Generalization comparison

The idea that we are going to explore in this part is the following one : since G-CNN are group equivariant they should be able to extract more information from the data than a regular CNN

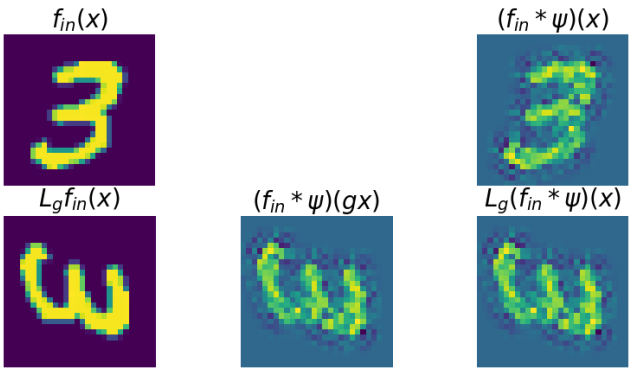

Figure 4: Verification of rotation equivariance in a P4CNN: applying the rotation before or after convolution yields the same transformed feature map.

5.1 Generalization Under Reduced Data Regimes

A central motivation for incorporating group equivariance into convolutional architectures is the resulting improvement in sample efficiency. Because each filter ψ is internally tied to its rotated (or reflected) counterparts through the group action, a G-CNN does not have to independently learn features appearing in multiple orientations. As discussed previously, the equivariance constraint eliminates redundant degrees of freedom in the learned representations, hence increasing parameter efficiency and accelerating convergence. This effect is already visible in Figure 3, where, on Rotated MNIST, models with larger equivariant groups achieve lower validation loss significantly earlier during training.

To quantify this gain in sample efficiency more explicitly, we study the generalization capability of the smallest G-CNN architecture compared with a baseline CNN of identical depth and comparable number of parameters. Our goal is to examine how performance deteriorates when the training set size is progressively reduced. Since G-CNNs can internally reuse each training image across all discrete group elements, one expects them to degrade much more slowly than standard CNNs.

Experimental protocol

We consider the standard MNIST dataset and construct four reduced training subsets containing 25 000, 10 000, 1 000, and 100 stratified samples, in addition to the full dataset. Both models are trained from scratch on each of these subsets with identical optimization hyperparameters. For every subset, we measure the final test accuracy after convergence.

Let f_θ denote the baseline CNN and $\tilde{f}_{\tilde{\theta}}$ the P_4-equivariant G-CNN. Because it enforces

$$\tilde{f}_{\tilde{\theta}}(L_g x) = L'_g \tilde{f}_{\tilde{\theta}}(x), \qquad g \in P_4,$$

each sample x implicitly provides four oriented training examples $\{L_g x\}_{g \in P_4}$ without augmenting the dataset or increasing the number of optimization steps. In contrast, the baseline CNN must learn these rotated versions independently from data. Therefore, if $\mathcal{D}_n$ denotes a dataset of size n, the effective number of informative pose variations seen by the equivariant model scales like $4n$, whereas for the standard CNN it remains n unless explicit augmentation is introduced.

Results.

The results of this experiment are summarized in Figure 5. While both models achieve comparable accuracy on the full dataset, the difference becomes increasingly pronounced as the number of training samples decreases. The G-CNN maintains above 90% accuracy even with only 1 000 samples, whereas the baseline CNN falls below 80%. At the extreme low-data regime of 100 samples, the performance gap widens dramatically: the baseline collapses toward 10% accuracy (random guessing), while the G-CNN retains nearly 20% accuracy, which is still low in an absolute sense, but almost twice the performance of the CNN under the same data constraints.

These results are consistent with the idea that an equivariant architecture has fewer patterns to learn from data. A standard CNN must discover from examples how a feature looks in every orientation, whereas a G-CNN already encodes this behavior in its structure. As the number of training samples becomes small, this built-in rotational knowledge plays a much larger role, preventing the model from overfitting to the limited examples it sees. Combined with the convergence analysis in Figure 3, these results demonstrate that G-CNNs not only train faster but also generalize significantly better in low data regimes.

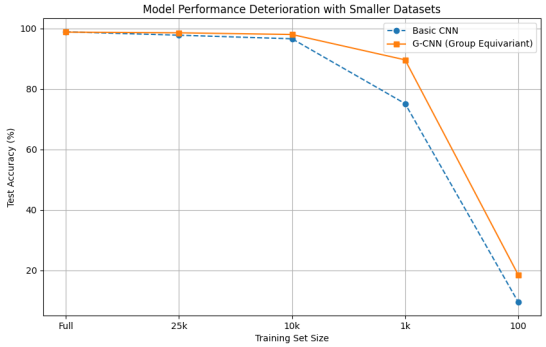

Figure 5: Test accuracy of a baseline CNN and a P_4-equivariant G-CNN as the MNIST training set size is progressively reduced. The G-CNN exhibits substantially slower deterioration, confirming the sample-efficiency advantage induced by equivariance.

6 Harmonic Networks

While P_n-equivariant CNNs provide built-in equivariance to discrete rotations, their symmetry group remains limited to n fixed orientations. This discretization becomes restrictive when the task requires robustness to arbitrary continuous rotations. To overcome this limitation, Harmonic Networks [5] extend the idea of group-equivariant CNNs to the continuous rotation group $SO(2)$ by designing filters that transform predictably under any rotation.

The core idea is to construct complex-valued convolutional filters with a predefined *rotation order* $\Delta m \in \mathbb{Z}$. When such a filter $W_{\Delta m}$ is convolved with a rotated input $R_\theta F$, the output transforms according to:

$$W_{\Delta m} * (R_\theta F) = e^{i\Delta m\theta}(W_{\Delta m} * F).$$

Thus, each filter produces a feature map whose phase encodes the rotation of the input. Stacking convolutions adds rotation orders, and if the sum of orders along any computational path is zero, the resulting feature is fully rotation-invariant.

This mechanism generalizes the notion of equivariance in G-CNNs: instead of encoding symmetry by sampling n discrete rotations (P_n), harmonic filters encode symmetry analytically through their complex phase structure.

6.1 Harmonic Convolution Layers

A harmonic convolutional layer consists of several *streams*, each corresponding to a specific rotation order. Filters between streams are constructed with rotation order differences:

$$\Delta m = m_{\text{out}} - m_{\text{in}}.$$

Convolutions between streams use complex-valued filters and are implemented using real-valued convolutions by decomposing:

$$(W*F)_{\mathbb{C}} = (\text{Re}\, W*\text{Re}\, F - \text{Im}\, W*\text{Im}\, F) + i(\text{Re}\, W*\text{Im}\, F + \text{Im}\, W*\text{Re}\, F).$$

Feature maps within each stream remain equivariant to continuous rotations, and the invariant stream ($\Delta m = 0$) provides rotation-invariant representations.

6.2 Constructing Rotation-Equivariant Filters

To ensure continuous rotational equivariance, each filter is expressed in polar coordinates as:

$$W_{\Delta m}(r, \phi) = R(r)\, e^{i(\Delta m\, \phi + \beta)},$$

where:

- $R(r)$ is a learnable radial profile, represented via a truncated Fourier series to guarantee smoothness,
- Δm controls the angular structure and ensures equivariance,
- β is a learned phase offset.

These continuous filters are sampled on a small grid (5×5) before convolution, ensuring computational efficiency compatible with standard CNN frameworks.

We analyzed the learned harmonic filters by examining the real and imaginary components for different values of $\Delta m = 0, 1, 2$. The results are consistent with the expected behavior of rotation-equivariant filters.

- $\Delta m = 0$**:** The filter exhibits an isotropic pattern, resembling a square in both the real and imaginary parts. This corresponds to the zero-order component, which is invariant to rotations and captures general intensity information.
- $\Delta m = 1$**:** The filters show a more irregular pattern with less obvious structure. This is characteristic of first-order components, which are sensitive to simple directional changes and capture edges or gradients along specific orientations.
- $\Delta m = 2$**:** The real part shows a vertical stripe, while the imaginary part forms a cross-like pattern. These second-order components detect more complex rotational structures, including diagonals and orientation-specific motifs. The separation of real and imaginary parts allows the network to represent both the magnitude and phase of oriented patterns.

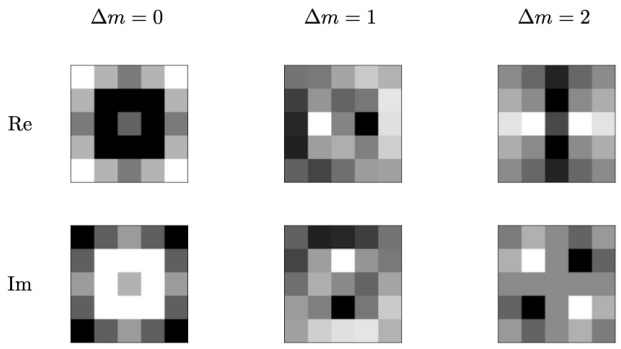

Figure 6: Examples of what the filters may look like after training a harmonic network

Overall, as Δm increases, the filters capture increasingly complex directional and rotational features, consistent with the design of harmonic convolutional networks.

6.3 Nonlinearities and Normalization

Standard pointwise nonlinearities (ReLU) do not preserve equivariance for complex-valued features. Harmonic Networks use magnitude-preserving complex nonlinearities, such as the complex shifted ReLU:

$$\text{ReLU}_b(re^{i\theta}) = \text{ReLU}(r + b)\, e^{i\theta}.$$

Batch normalization is applied to magnitudes only, ensuring that normalization layers do not break the equivariant structure.

6.4 Experiments

To evaluate the rotational equivariance of Harmonic Networks, we perform the same set of experiments as in Section 4, but using the Harmonic Network architecture on the same datasets.

Table 2: Test accuracy (%) of P4CNN and Harmonic Networks (HN) on different datasets.

Dataset	P4CNN	HN - 2 streams	HN - 3 streams	HN - Worrall et al.
MNIST	99.0	99.2	99.4	**99.7**
Rotated MNIST	96.0	98.3	98.29	**98.31**
CIFAR-10	83.0	84.7	**85.8**	-
Augmented CIFAR-10	84.2	84.7	**85.3**	-

We can see in Table 2 that increasing the number of streams in the Harmonic Network generally leads to slightly better performance. Using $m = 3$ streams allows the network to capture richer rotational features, resulting small gains compared to the 2-stream version.

7 Conclusion

In this report, we have explored the concept of equivariance in convolutional neural networks beyond standard translations. We demonstrated that it is possible and beneficial to constrain CNNs to be equivariant to additional transformations such as rotations, both in a discrete manner (as in P4M CNNs) and in a continuous manner (as in Harmonic Networks).

Our experiments on standard datasets show that incorporating rotational equivariance leads to consistent improvements in classification performance. In particular, Harmonic Networks with $m = 3$ streams generally achieve slightly higher accuracy compared to 2-stream versions. These results confirm the theoretical advantages of designing CNNs with structured equivariance.

Overall, while standard translation-equivariant CNNs remain effective, extending equivariance to other transformations provides a powerful tool to improve performance in scenarios where data exhibits symmetries. This is a very active area of research, with numerous follow-up works such as Steerable CNNs [2], 3D Steerable CNNs [4], and Learning to Convolve [3].

References

[1] Taco S. Cohen and Max Welling. 2016. Group Equivariant Convolutional Networks. arXiv:1602.07576 [cs.LG] https://arxiv.org/abs/1602.07576

[2] Taco S. Cohen and Max Welling. 2016. Steerable CNNs. arXiv:1612.08498 [cs.LG] https://arxiv.org/abs/1612.08498

[3] Nichita Diaconu and Daniel E Worrall. 2019. Learning to Convolve: A Generalized Weight-Tying Approach. arXiv:1905.04663 [cs.LG] https://arxiv.org/abs/1905.04663

[4] Maurice Weiler, Mario Geiger, Max Welling, Wouter Boomsma, and Taco Cohen. 2018. 3D Steerable CNNs: Learning Rotationally Equivariant Features in Volumetric Data. arXiv:1807.02547 [cs.LG] https://arxiv.org/abs/1807.02547

[5] Daniel E. Worrall, Stephan J. Garbin, Daniyar Turmukhambetov, and Gabriel J. Brostow. 2017. Harmonic Networks: Deep Translation and Rotation Equivariance. arXiv:1612.04642 [cs.CV] https://arxiv.org/abs/1612.04642

A Proof of Translation Equivariance in CNNs

We want to demonstrate that, for any translation t, the following two operations are equivalent:

- Translate the input f by t and then convolve with the filter ψ.
- Convolve f with ψ first, and then translate the resulting feature map by t.

Our proof proceeds slightly differently from the original paper. By performing the substitution $y \mapsto x + y$, the convolution can be rewritten as:

$$(f \star \psi)(x) = \sum_{y \in \mathbb{Z}^2} f(y)\, \psi(x - y) = \sum_{y \in \mathbb{Z}^2} f(x + y)\, \psi(y).$$

We now examine both sides of the equivalence separately. First, consider $(L_t f) \star \psi$, where $L_t f = f \circ t^{-1}$:

$$(L_t f \star \psi)(x) = \sum_{y \in \mathbb{Z}^2} f(t^{-1}(x + y))\, \psi(y) = \sum_{y \in \mathbb{Z}^2} f(x + y - t)\, \psi(y).$$

Next, consider $L_t(f \star \psi)$:

$$(L_t(f \star \psi))(x) = (f \star \psi)(x - t) = \sum_{y \in \mathbb{Z}^2} f((x - t) + y)\, \psi(y) = \sum_{y \in \mathbb{Z}^2} f(x + y - t)\, \psi(y).$$

Since both expressions are identical, the translation operator commutes with the convolution, confirming translation equivariance. Intuitively, in standard convolution, each term in the sum either transforms the input with a fixed filter or transforms the filter with a fixed input; in both cases, the result is identical.

Extending this notion to general groups G, the inputs to $f \star \psi$ are now elements of G rather than pixels. Although reasoning about arbitrary groups can be challenging, our setting is straightforward: for translations, $T \simeq \mathbb{Z}^2$, so functions on the group resemble conventional images.

To allow recursive convolution with multiple filters, we adopt the definition of *group convolution*:

$$(f \star \psi)(g) = \sum_{h \in G} f(gh)\, \psi(h),$$

where $g, h \in G$ and gh denotes the group composition. This construction is equivariant with respect to G, analogous to standard CNNs. The substitution $h \mapsto gh$ is the key step ensuring equivariance in the original derivation.

B Proof of Rotation Equivariance in Harmonic Networks

We now demonstrate that Harmonic Networks are equivariant to discrete rotations. Let R_θ denote a rotation by angle $\theta \in \{0, \frac{2\pi}{M}, \ldots, \frac{2\pi(M-1)}{M}\}$, and consider an input f and a circular harmonic filter ψ_m of order m. The circular harmonic convolution is defined as:

$$(f \star \psi_m)(x) = \sum_{y \in \mathbb{Z}^2} f(y)\, \psi_m(x - y),$$

with the filter having the property:

$$\psi_m(R_\theta y) = e^{im\theta}\, \psi_m(y).$$

First, consider rotating the input before convolution:

$$(L_{R_\theta} f \star \psi_m)(x) = \sum_{y \in \mathbb{Z}^2} f(R_\theta^{-1}(x + y))\, \psi_m(y).$$

Performing the substitution $y \mapsto R_\theta y$ gives:

$$(L_{R_\theta} f \star \psi_m)(x) = \sum_{y \in \mathbb{Z}^2} f(x + y)\, \psi_m(R_\theta y) = \sum_{y \in \mathbb{Z}^2} f(x + y)\, e^{im\theta} \psi_m(y) = e^{im\theta}\, (f \star \psi_m)(x).$$

Next, consider applying the rotation to the feature map after convolution:

$$L_{R_\theta}(f \star \psi_m)(x) = (f \star \psi_m)(R_\theta^{-1} x) = \sum_{y \in \mathbb{Z}^2} f(R_\theta^{-1} x + y)\, \psi_m(y) = e^{im\theta}\, (f \star \psi_m)(x),$$

where we again used the harmonic property of the filter. Since both operations yield the same result up to the phase factor $e^{im\theta}$, the convolution is equivariant to rotations. Intuitively, the circular harmonic filter rotates in the complex plane according to its order m.