Symmetries in ML models and Group Equivariance

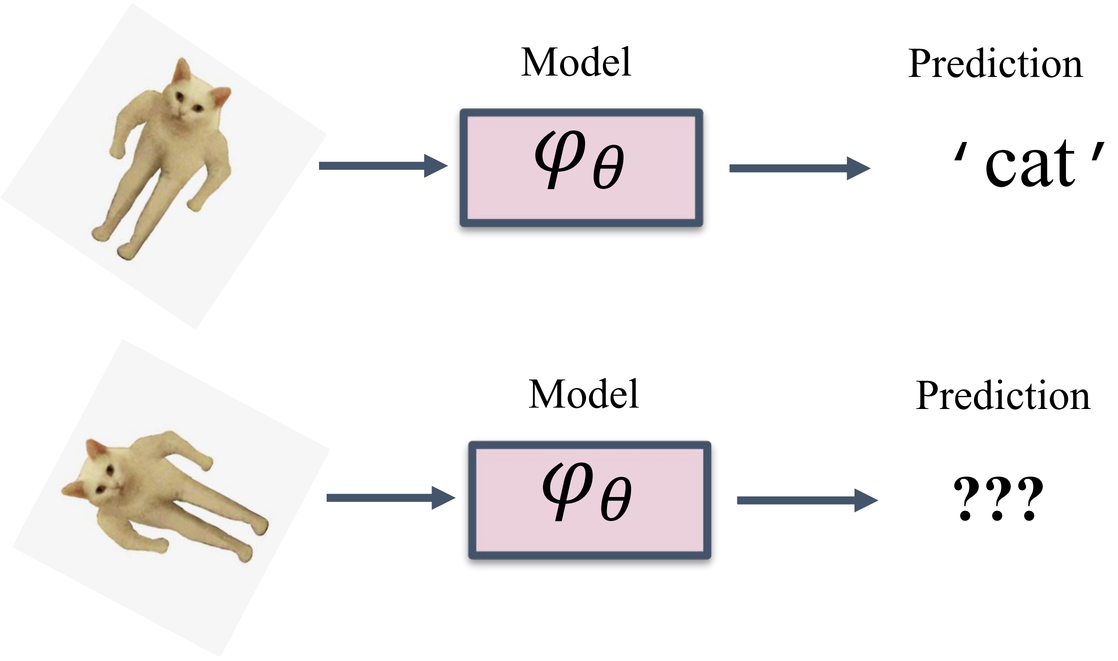
Wenhan Gao

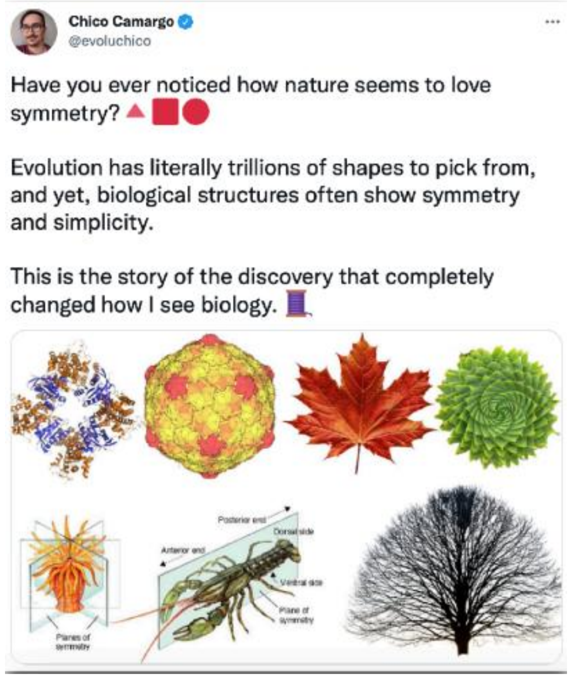
AMS, Stony Brook University

Introduction

Group equivariance in ML models is about **enforcing symmetries** in our architectures.

- Many learning tasks, oftentimes, **have symmetries** under some set of transformations acting on the data.
- More importantly, **the nature itself is about symmetries**.

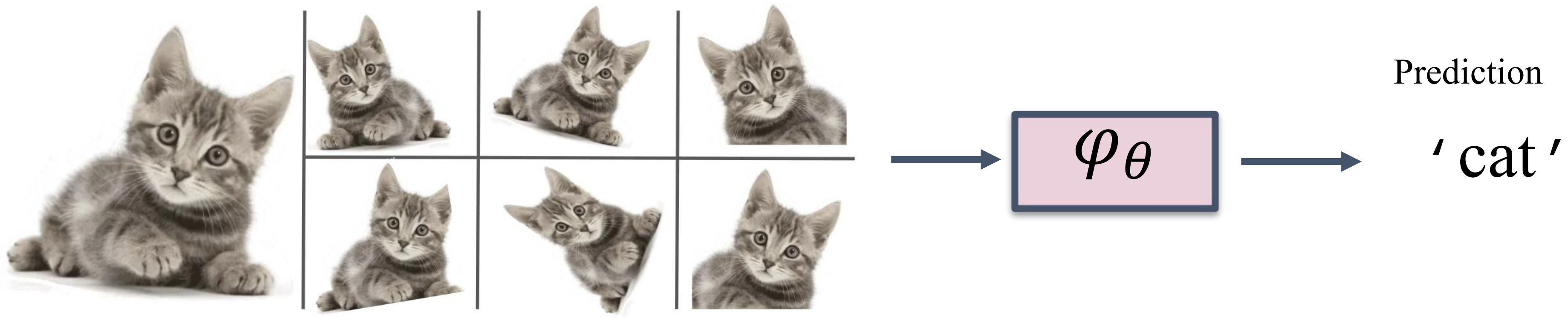




FYI: Dr. Chen Ning Yang received the Nobel Prize in physics (1957) for discoveries about symmetries, and his B.S. thesis is "Group Theory and Molecular Spectra".

Introduction: Learning Symmetries

To **learn symmetry**, a common approach is to do *data-augmentation*: Feed *augmented* data and **hope** the model "learns" the symmetry.



Issues:

- ✗ **No guarantee** of having symmetries in the model
- ✗ **Wasting valuable net capacity** on learning symmetries from data
- ✗ **Redundancy** in learned feature representation

Solution:

- ✓ Building symmetries into the model by design!

Introduction: Group Equivariant Models

What is group equivariance? In a nutshell, group equivariance means that **if the input is transformed, the output will be changed in a predictable way.**

Building symmetry into ML has led to major breakthroughs in deep learning:

- Imposing **translational symmetry** and parameter sharing allowed CNNs to essentially solve computer vision.
- Group Equivariance **conceptualizes CNN success (symmetry exploitation) and generalizes it to tasks with other symmetries.**

In this talk, I will introduce group equivariance and group equivariant architectures. **This talk will cover the following topics** with a focus on intuition over mathematical rigors:

❖ Group Theory Mathematical Preliminary

❖ CNNs and Translation Equivariance

❖ From CNNs to Regular Group CNNs for SE(2) Equivariance

❖ Benefits of Group CNNs

Mathematical Preliminary: Group

A **group** $(G, \cdot)$ is a set of elements G equipped with a group product $\cdot$, a binary operator, that satisfies the following four axioms:

- Closure: Given two elements g and h of G, the product $g \cdot h$ is also in G.
- Associativity: For $g, h, i \in G$ the product $\cdot$ is associative, i.e., $g \cdot (h \cdot i) = (g \cdot h) \cdot i$.
- Identity element: There exists an identity element $e \in G$ such that $e \cdot g = g \cdot e = g$ for any $g \in G$.
- Inverse element: For each $g \in G$ there exists an inverse element $g^{-1} \in G$ s.t. $g^{-1} \cdot g = g \cdot g^{-1} = e$.

Example:

The translation group consists of all possible translations in $\mathbb{R}^2$ and is equipped with the group product and group inverse:

$$
\begin{aligned}
g \cdot g' &= (t + t'), \quad t, t' \in \mathbb{R}^2 \\
g^{-1} &= (-t),
\end{aligned}
$$

with $g = (t), g' = (t')$, and $e = (0, 0)$.

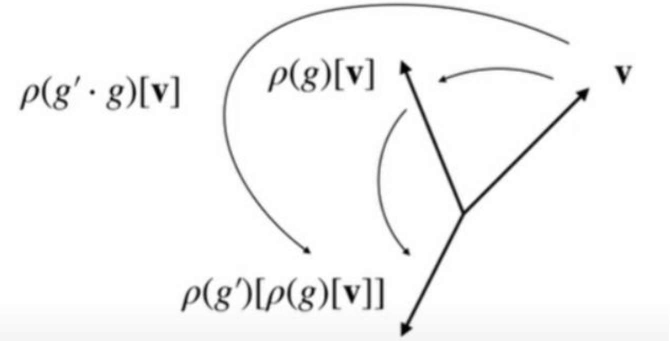
In plain English:

- Closure: when you apply two transformations of interest, it is still a transformation of interest.
- Associativity: the order of "grouping" transformations does not matter.
- Identity: we should be able to not apply any transformation at all.
- Inverse: we should be able to transform back.

Mathematical Preliminary: Representation

A **representation** $\rho : G \rightarrow GL(V)$ is a group homomorphism from G to the general linear group $GL(V)$. That is, $\rho(g)$ is a linear transformation parameterized by group elements $g \in G$ that transforms some vector $\mathbf{v} \in V$ (e.g. an image) such that

$$\rho\left(g'\right) \circ \rho(g)[\mathbf{v}] = \rho\left(g' \cdot g\right)[\mathbf{v}]$$



This essentially means that we can transfer group structure to other types of objects now, such as vectors or images.

Note: A **homomorphism** is a structure-preserving map between two algebraic structures of the same type (such as two groups, two rings, or two vector spaces). A general linear group is the group of all invertible $d_V \times d_V$ matrices.

A **left-regular representation** $\mathscr{L}_g$ is a representation that transforms functions f by transforming their domains via the inverse group action

$$\mathscr{L}_g[f](x) := f\left(g^{-1} \cdot x\right)$$

Image Source: https://uvagedl.github.io/

Mathematical Preliminary: Representation

Example I:

1. $f \in \mathbb{L}_2(\mathbb{R})$: A function defined on a line.
2. $G = \mathbb{R}$: The 1D translation group.
3. $[\mathscr{L}_{g=t} f](x) = f\left(t^{-1} \odot x\right) = f(x-t)$: A translation of the function.

Example II:

1. $f \in \mathbb{L}_2\left(\mathbb{R}^2\right)$: A 2D image.
2. $G = SE(2)$: The 2D roto-translation group.
3. $[\mathscr{L}_{g=(t,\theta)} f](\mathbf{x}) = f\left(\mathbf{R}_\theta^{-1}(x-t)\right)$: A roto-translation of the image.

Remark: Group Structure on Different Objects

1. Group Product (acting on G it self): $g \cdot g'$
2. Left Regular Representation (acting on a vector spaces): $\mathscr{L}_g f$
3. Group Actions (acting on $\mathbb{R}^d$): $g \odot x$

Mathematical Preliminary: Equivariance and Invariance

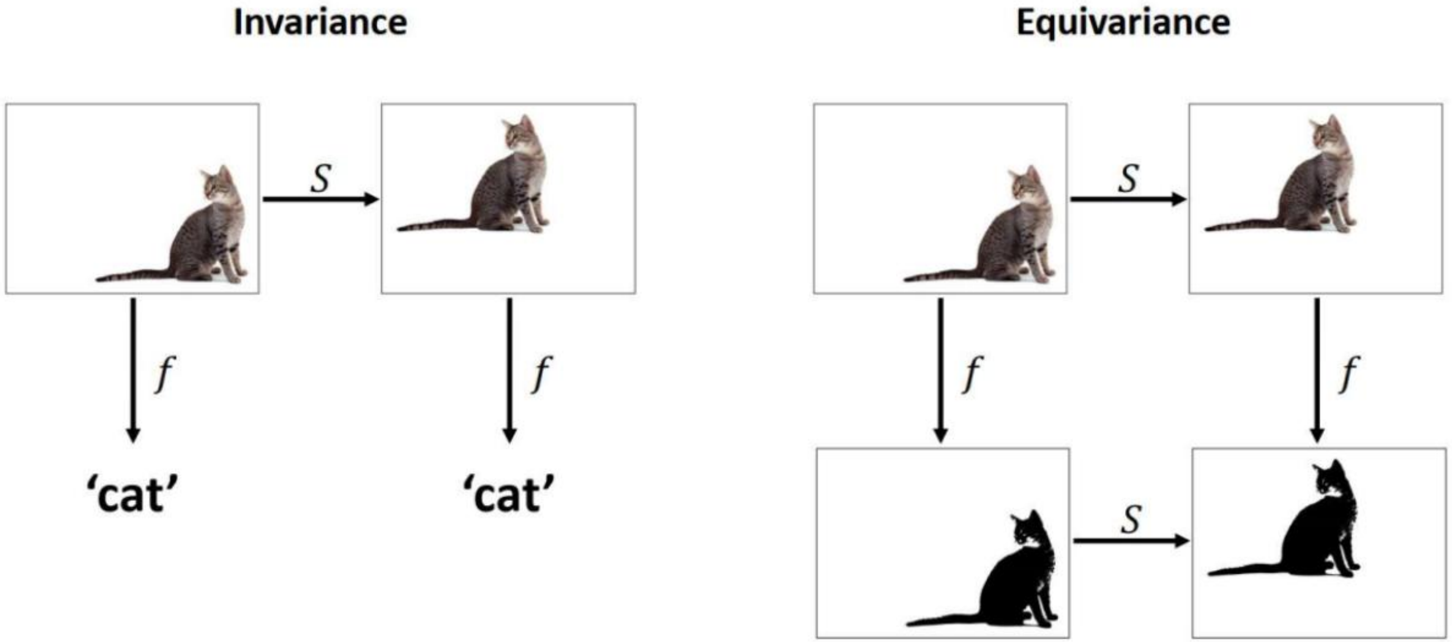
Equivariance is a property of an operator $\Phi : X \to Y$ (such as a neural network layer) by which it commutes with the group action:

$$\Phi \circ \rho^X(g) = \rho^Y(g) \circ \Phi,$$

Invariance is a property of an operator $\Phi : X \to Y$ (such as a neural network layer) by which it remains unchanged after the group action:

$$\Phi \circ \rho^X(g) = \Phi,$$

- $\rho^X(g)$: group representation action on X
- $\rho^Y(g)$: group representation action on Y
- Invariance is a special case of equivariance when $\rho^Y(g)$ is the identity.



CNNs and Translation Equivariance: Convolution, Cross-Correlation

Definition (Convolution):

> The convolution of f and g is written as $f * g$, denoting the operator with the symbol $*$. It is defined as the integral of the product of the two functions after one is reflected and shifted. As such, it is a particular kind of integral transform:
> $$(k * f)(x) := \int_{\mathbb{R}^d} k(x - x')f(x')dx'.$$

> An equivalent definition is (commutativity):
> $$(k * f)(x) := \int_{\mathbb{R}^d} k(x')f(x - x')dx'.$$

Definition (Cross-Correlation):

> The corss-correlation of f and g is written $f \star g$, denoting the operator with the symbol $\star$. It is defined as the integral of the product of the two functions after one is shifted. As such, it is a particular kind of integral transform:
> $$(k \star f)(x) := \int_{\mathbb{R}^d} k(x' - x)f(x')dx'.$$
> An equivalent definition is (not commutativity in this case):
> $$(k \star f)(x) := \int_{\mathbb{R}^d} k(x')f(x' + x)dx'.$$

As a convention, we actually perform cross-correlation in CNNs. If a CNN can learn a task using convolution operation, it can also learn the same task using correlation operation (It would learn the rotated, along the diagonal, version of each filter).

CNNs and Translation Equivariance: Translation Equivariance

Convolution and Cross-Correlation are translation equivariant, so are their discrete counterparts.

Example:

1. Translate f by t first, then apply the convolution:

$$(k \star \mathscr{L}_t f)(x) = \int_{\mathbb{R}^d} k(x' - x)[t^{-1} \odot f(x')]dx' = \int_{\mathbb{R}^d} k(x' - x) f(x' - t) dx'.$$

2. Apply convolution first, and then translate by t:

$$\mathscr{L}_t (k \star f)(x) = \mathscr{L}_t \int_{\mathbb{R}^d} k(x' - (x - t)) f(x') dx' = \int_{\mathbb{R}^d} k(x' - x + t) f(x') dx' = \int_{\mathbb{R}^d} k(x' - x) f(x' - t) dx.$$

In the last equality, we just replace x' by $x' - t$. Note that this operation is valid because this substitution is a bijection $\mathbb{R}^d \to \mathbb{R}^d$ and we integrate over the entire $\mathbb{R}^d$.

By similar arguments, we can prove translation equivariance for convolution and the discrete versions.

CNNs and Translation Equivariance: Intuition

Mathematically, it is easy to prove translation equivariance. However, let's look at the definiton of cross-correlation again to gain some intution about how to achieve equivariance.

Cross-Correlation:

$$(k \star f)(x) := \int_{\mathbb{R}^d} k(x' - x) f(x') dx'.$$

Replace x' by $x' + x$:

$$(k \star f)(x) := \int_{\mathbb{R}^d} k(x') f(x' + x) dx'.$$

Intuition: $f(x' + x)$ represents a translated version of $f(x)$. We have created many translated version of $f(x)$ while creating the feature map. If we need to compute the cross-correlation for a transformed f, we can just go and look up the relevant outputs, because we have already computed them. Equivalently, $k(x' - x)$ represents a translated version of $k(x)$.

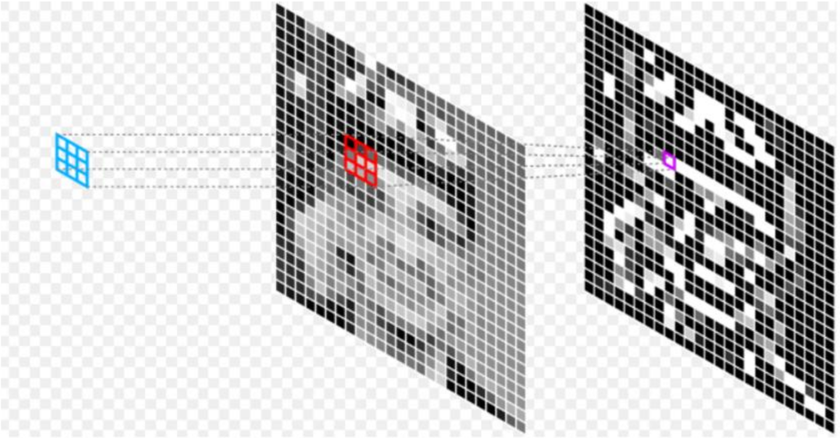


Image Source: StackOverflow

CNNs and Translation Equivariance: Generalization

$$(k \star f)(x) := \int_{\mathbb{R}^d} k(x'-x) f(x') dx' = \int_{\mathbb{R}^d} [\mathscr{L}_x k(x')] f(x') dx' = \langle \mathscr{L}_x k, f \rangle_{\mathbb{L}_2(\mathbb{R}^d)} = \langle k, \mathscr{L}_{-x} f \rangle_{\mathbb{L}_2(\mathbb{R}^d)}.$$

The domain is the translation group

Integral over the Translation Group

Translated Kernels

Here, we explicitly think of the cross-correlation in terms of translations. To generalize, if we want to transform f with other groups, the trick is to make the kernel k to be represented by a group. Group representations on k is reflected on f as well.

To generalize to other groups, we should consider the follwoings:

- Make the function defined on the group of interest.
- Integrate over the group of interest
- Make the kernel reflect the actions of the group of interest

Regular Group CNN and SE(2) Equivariance: SE(2) Lifting Correlation

- To make the function defined on the group of interest, we define the lifting operation.

The lifting correlation of f and g is written $f \star_{SE(2)} g$, denoting the operator with the symbol $\star_{SE(2)}$. It is defined as the integral of the product of the two functions after one is shifted and rotated. As such, it is a particular kind of integral transform:

$$(k \star_{SE(2)} f)(x,\theta) := \int_{\mathbb{R}^2} k\left(\mathbf{R}_\theta^{-1}(x'-x)\right) f(x')dx' = \int_{\mathbb{R}^2} [\mathscr{L}_{g=(x,\theta)} k(x')] f(x')dx' = \langle \mathscr{L}_{g=(x,\theta)} k, f\rangle_{\mathbb{L}_2(\mathbb{R}^2)}.$$

Lifting correlation raise the feature map to a higher dimension that represents rotation. Now planar rotation becomes a rotation in xy-axes and a periodic shift in θ-axis.

$$(k \star_{SE(2)} f)(x,\theta) := \int_{\mathbb{R}^2} k\left(\mathbf{R}_\theta^{-1}(x'-x)\right) f(x')dx' = \int_{\mathbb{R}^2} [\mathscr{L}_{g=(x,\theta)} k(x')] f(x')dx' = \langle \mathscr{L}_{g=(x,\theta)} k, f\rangle_{\mathbb{L}_2(\mathbb{R}^2)}.$$

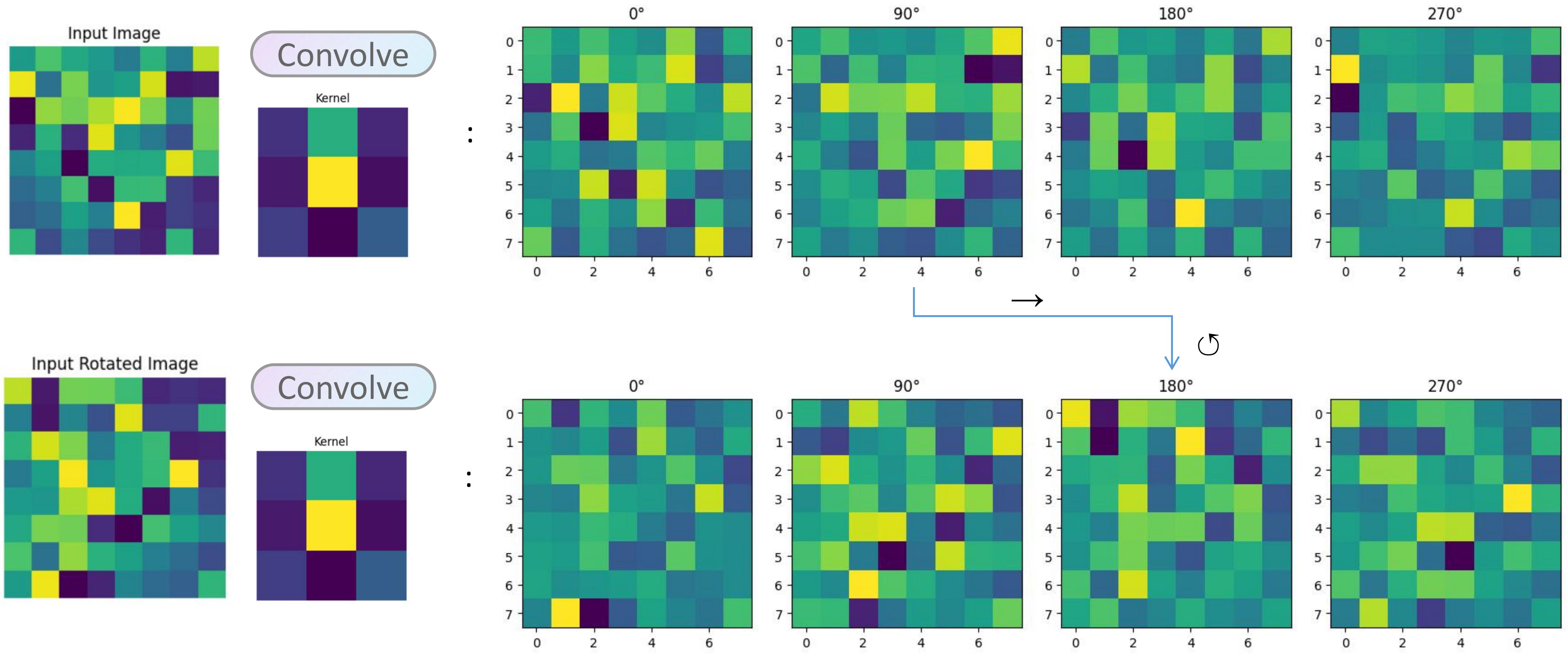
The domain of the output is lifted to the SE(2) group

Still R(2) for lifting operation because the function is define on R(2)

The kernel reflects the SE(2) group now

The domain of the input is R(2)

Regular Group CNN and SE(2) Equivariance: SE(2) Lifting Correlation



(Rotate the Input) ↺ = ⟶ + ↺ (Periodic Shift + Planar Rotation for the Output)

Regular Group CNN and SE(2) Equivariance: SE(2) Cross Correlation

Now, the function is already defined on the group of interest, we still need to:

- Integrate over the group of interest
- Make the kernel reflect the actions of the group of interest

The group correlation of f and g is written $f \star_{SE(2)} g$, denoting the operator with the symbol $\star_{SE(2)}$. It is defined as the integral of the product of the two functions after one is shifted and rotated. As such, it is a particular kind of integral transform:

$$
\begin{aligned}
(k \star_{SE(2)} f)(x,\theta) :&= \int_{\mathbb{SE}(2)} k\Big(\mathbf{R}_\theta^{-1}(x'-x), \theta' - \theta \mod 2\pi\Big) f(x',\theta') d\theta' dx' \\
&= \int_{\mathbb{SE}(2)} [\mathscr{L}_{g=(x,\theta)} k(x',\theta')] f(x',\theta') d\theta' dx' \\
&= \langle \mathscr{L}_{g=(x,\theta)} k, f \rangle_{\mathbb{L}_2(\mathbb{SE}2))}.
\end{aligned}
$$

Since $SE(2)$ is a semidirect group, $SE(2) = \mathbb{R}^2 \ltimes SO(2)$, another way to view this operation is to split roto-translation into rotation plus translation:

$$
(k \star_{SE(2)} f)(x,\theta) := \langle \mathscr{L}_{g=(x,\theta)} k, f \rangle_{\mathbb{L}_2(\mathbb{SE}(2))} = \langle \mathscr{L}_{g=x} \mathscr{L}_{g=\theta} k, f \rangle_{\mathbb{L}_2(\mathbb{SE}(2))}.
$$

So basically, what happens in group correlation is: We have rotated plannar kernels cross-correlate with each of the input planar features ($\mathbf{R}_\theta^{-1}(x'-x)$ comes in here), and then we also have cross-correlation with the rotation part ($(\theta' - \theta)$ comes in here); in other words, each kernels go through a conv operator for the planar part and then mixed with other kernels with different weights (1D conv on the rotation dimension). Therefore, we have equivariance for both plannar part and rotation part.

Regular Group CNN and SE(2) Equivariance: SE(2) Cross Correlation

$$(k \star_{SE(2)} f)(x,\theta) := \int_{\mathbb{R}^2} k\left(\mathbf{R}_\theta^{-1}(x'-x)\right) f(x')dx'$$

The domain of the output is lifted to the SE(2) group

Still R(2) for lifting operation because the function is define on R(2)

Planar Rotation to raise to the feature maps to the group SE(2)

The domain of the input is R(2)

$$(k \star_{SE(2)} f)(x,\theta) := \int_{SE(2)} k\left(\mathbf{R}_\theta^{-1}(x'-x), \theta'-\theta \mod 2\pi\right) f(x',\theta')\, d\theta'\, dx'$$

The domain of the output is still the SE(2) group

Integrate over SE(2) because the function is now define on SE(2)

Planar Rotation

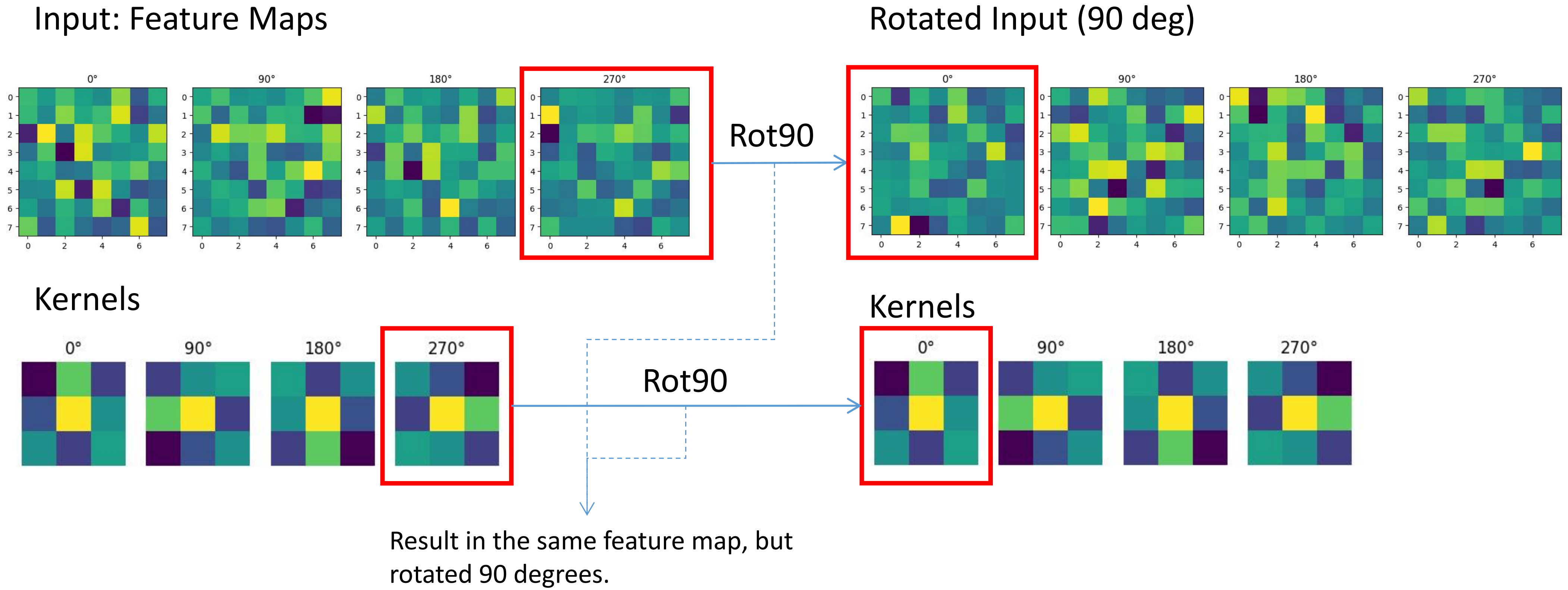
Periodic Shift

The domain of the input is SE(2)

Reflect the SE(2) Group

Regular Group CNN and SE(2) Equivariance: SE(2) Cross Correlation

The goal is still
(Rotate the Input) ↺ = ⟶ + ↺ (Periodic Shift + Planar Rotation for the Output)



Thus, the resulting feature maps will still be rotated and periodically shifted. It seems that so far, we only used $\mathbf{R}_\theta^{-1}(x' - x)$, but recall that, in group correlation, we also have $\theta' - \theta$. Now, imagine when the input is rotated 180 deg, the above equivariance does not hold anymore. That's why we actually need to have convolution on the theta axis as well.

Regular Group CNN and SE(2) Equivariance: More Intuition

Although the examples are given for the group SE(2), the idea can generalize to other affine groups (semi-direct product groups).

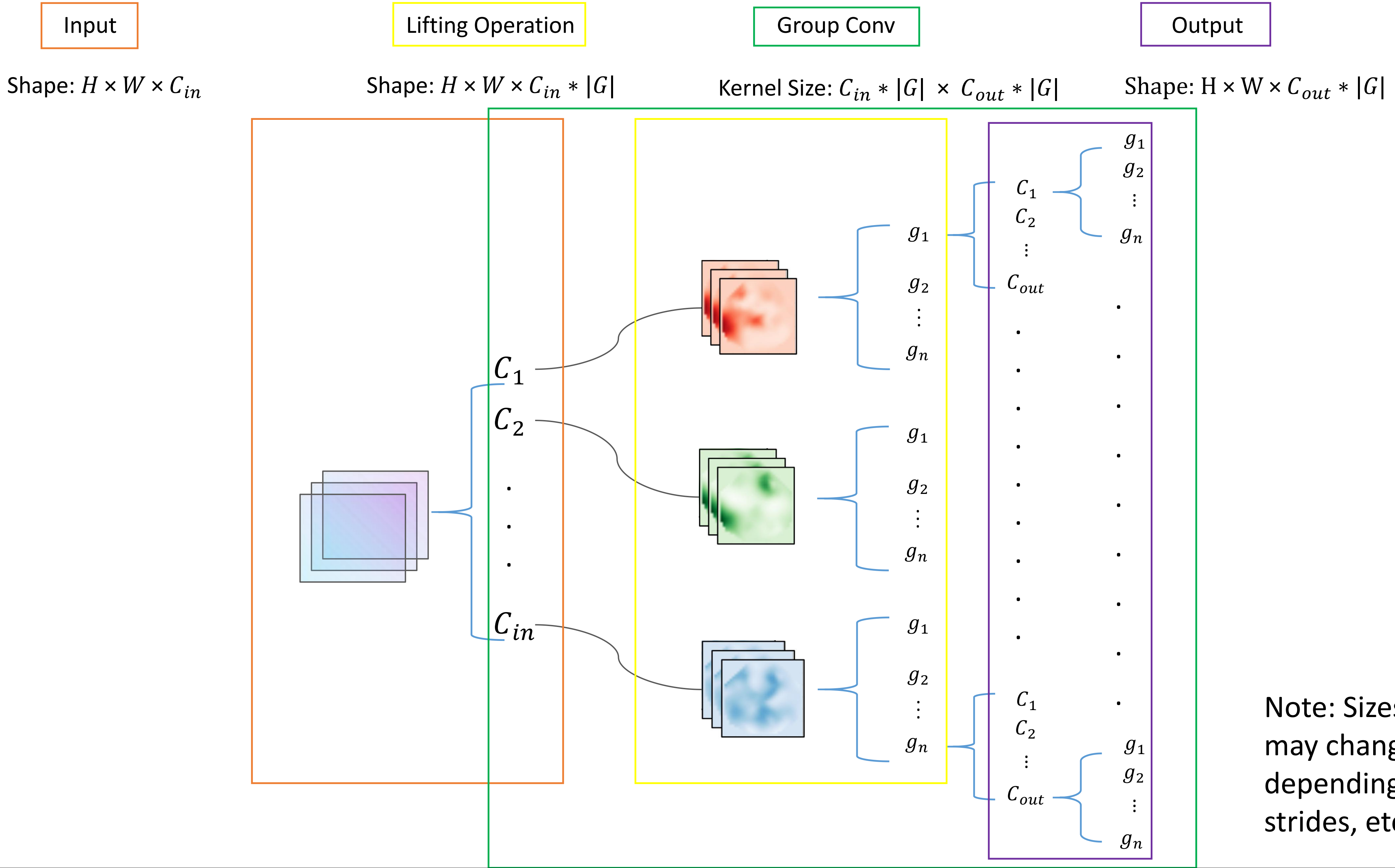
If we look carefully at how rotational equivariance is achieved, we find that it basically adds a rotation dimension represented by an axis θ, and thus, rotational equivariance problem now becomes translation equivariance problem which can be solved easily by 1D convolution/cross-correlation.

$$\text{translational weight sharing} \iff \text{translation group equivariance}$$
$$\text{affine weight sharing} \iff \text{affine group equivariance}$$

Note: Translations and H-transformations form so-called affine groups

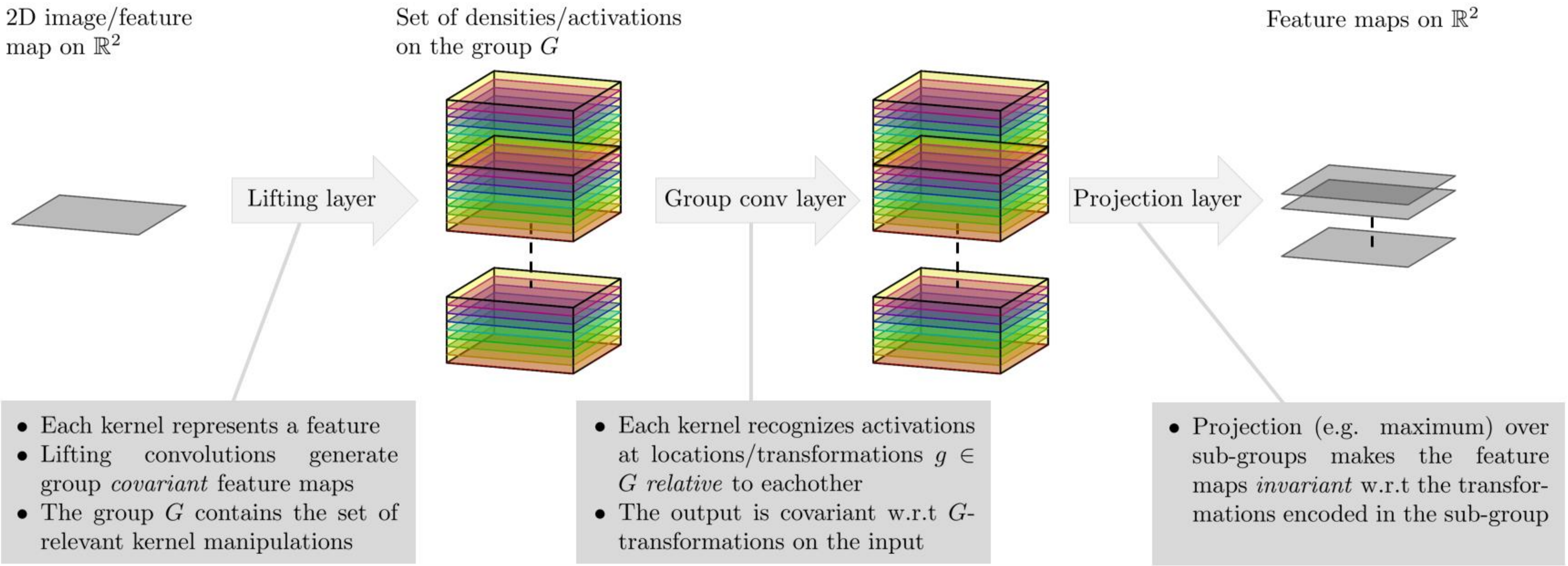
$$\text{Aff}(H) := (\mathbb{R}^d, +) \rtimes H.$$

An overview of actual implementation with nn.Conv2d()



Note: Sizes of H and W may change depending on padding, strides, etc..

Regular Group CNN and SE(2) Equivariance: Example



1. Lifting Layer (Generate group equivariant feature maps):
   - 2D input ⇒ 3D feature map with the third dimension being rotation.
2. Group Conv Layer (Group equivariant on the input):
   - 3D feature map ⇒ 3D feature map
3. Projection Layer:
   - Invariance: 3D feature map ⇒ 2D feature map by (e.g. max/avg) pooling over θ dimension. Now, it is invairant in θ dimension.
   - Equivariance: The resulting 2D feature map is rotation equivariant w.r.t. the input.

Image Source: https://uvagedl.github.io/

Results: Group Equivariant Convolutional Networks

Results on datasets with rotations: The rotated MNIST dataset contains 62000 randomly rotated handwritten digits.

Network	Test Error (%)
Larochelle et al. (2007)	10.38 ± 0.27
Sohn & Lee (2012)	4.2
Schmidt & Roth (2012)	3.98
Z2CNN	5.03 ± 0.0020
P4CNNRotationPooling	3.21 ± 0.0012
P4CNN	**2.28 ± 0.0004**

Table 1. Error rates on rotated MNIST (with standard deviation under variation of the random seed).

Z2CNN: Normal CNN

P4CNNRotation Pooling: P4CNN but impose rotation invariance in every layer

P4CNN: only rotation invariance for the output layer, equivariance for the intermediate layers.

p4: Cyclic rotation group of order 4 (0, 90, 180, 270)
p4m: p4 plus 4 horizontal and vertical flips

As expected, Group conv can ***improve model performance when (global)symmetries exists***.

Results on datasets without rotations: CIFAR10+: moderate data augmentation with horizontal flips and small translations

Network	G	CIFAR10	CIFAR10+	Param.
All-CNN	$\mathbb{Z}^2$	9.44	8.86	1.37M
	$p4$	8.84	7.67	1.37M
	$p4m$	7.59	7.04	1.22M
ResNet44	$\mathbb{Z}^2$	9.45	5.61	2.64M
	$p4m$	**6.46**	**4.94**	2.62M

Table 2. Comparison of conventional (i.e. $\mathbb{Z}^2$), $p4$ and $p4m$ CNNs on CIFAR10 and augmented CIFAR10+. Test set error rates and number of parameters are reported.

The CIFAR dataset is not actually symmetric, since objects typically appear upright. Nevertheless, we see substantial increases in accuracy on this dataset, indicating that there need not be a full symmetry for G-convolutions to be beneficial.
In the absence of global symmetries, Group Conv can still improve the performance due to its ability to capture local symmetries.

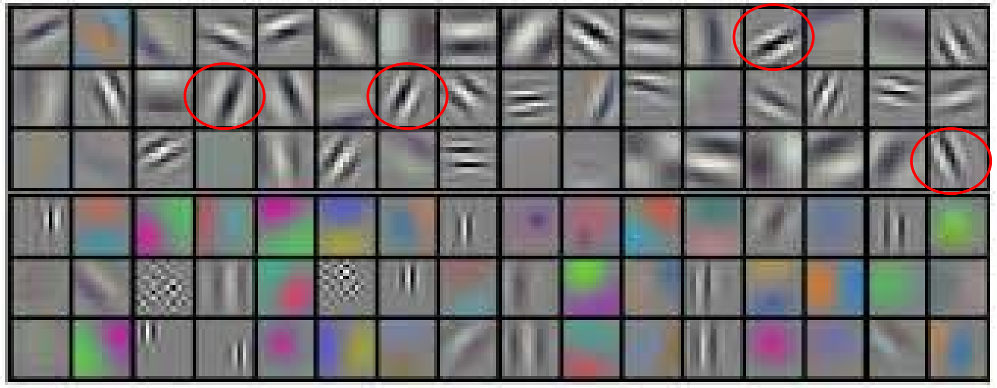
Group Equivariant Convolutional Networks; Taco S. Cohen, Max Welling ICML 2016 (https://arxiv.org/pdf/1602.07576.pdf)

Regular Group CNN: Intuition for Benefits and Advantages

The benefits of having equivariant NN architecture can be summarized as follows:

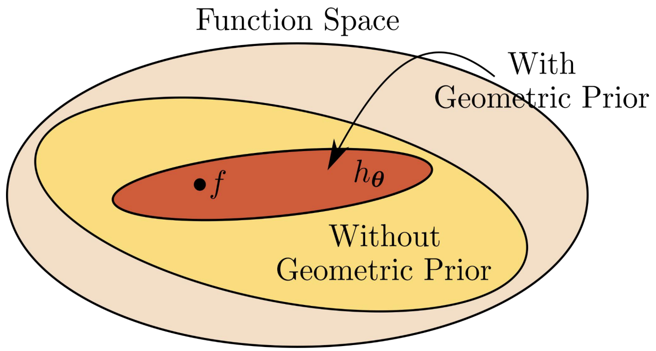
- ❑ **Equivariance**: We have geometric guarantee that the model is equivariant to certain symmetry groups.

- ❑ **Richer Feature Representations**:



Normal CNN kernel learns roto-translated features, but they are inherent in Group CNN.

- ❑ **Generalization and Efficient Learning:** Geometric priors constrain the parameter search space to smaller region → less parameters, better generalization with less data.



Conclusion

In this talk, we covered

- The issues in data augmentation to attain symmetries and the motivations of having symmetries in the model itself.
- Several basic mathematical concepts needed to understand group equivariance.
- Definitions of convolution and cross-correlation and the intuition why they are equivariant under translations.
- Generalization of the notions of translational equivariance in normal CNNs to building group equivariant CNNs.
- The mathematical formulations of group CNNs and the intuitions behind the mathematics.
- The results in the original group CNN paper.
- The intuition of the benefits of having equivariant models, which can be beyond simply achieving symmetries.