Group Equivariant Convolutional Networks Theory and Application

Shaoxuan Chen

Department of Mathematics and Statistics, UMass-Amherst

Outline

0. Deep Neural Network(DNN) and Convolution Neural Network(CNN)

1. General Group Theory

2. Group-equivariant CNNs.

3. Rotation-equivariant Steerable Filter CNNs

4. Scale-equivariant Steerable CNNs

5. Stability of equivariant representation to deformation

6. Experiment Results in Fashion-MNIST Dataset

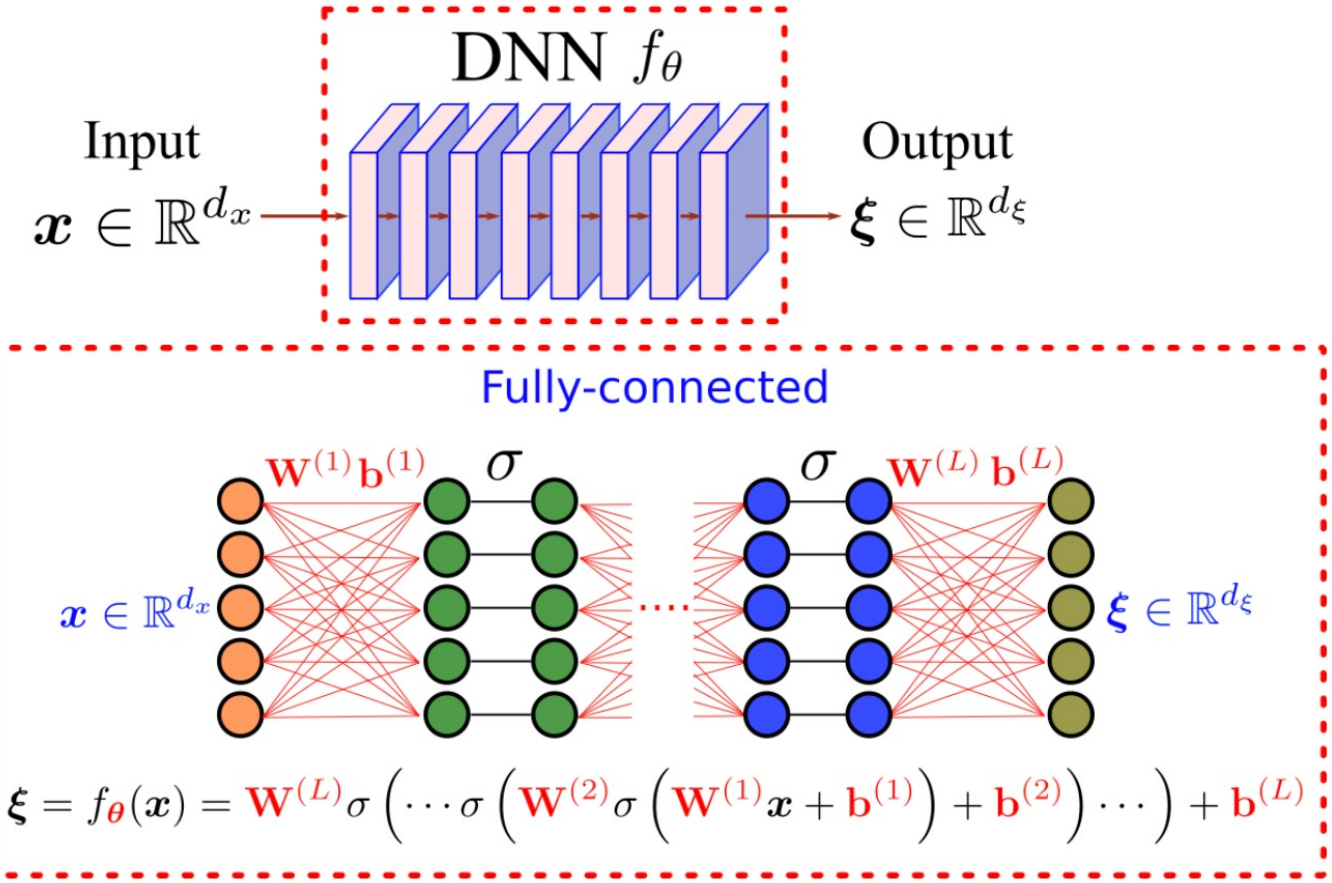
7. Future Directions

Related Papers

- Cohen, T. and Welling, M. Group Equivariant Convolutional Networks. ICML 2016.
- Weiler, M and Hamprecht, F A and Storath, M. Learning Steerable Filters for Rotation Equivariant CNNs. CVPR 2018.
- Sosnovik, I., Szmaja, M and Smeulders, A. Scale-Equivariant Steerable Networks, ICLR 2020
- Zhu, W and Qiu, Q and Calderbank, R and Sapiro, G and Cheng, X. Scaling-Translation-Equivariant Networks with Decomposed Convolutional Filters, JMLR 2022.

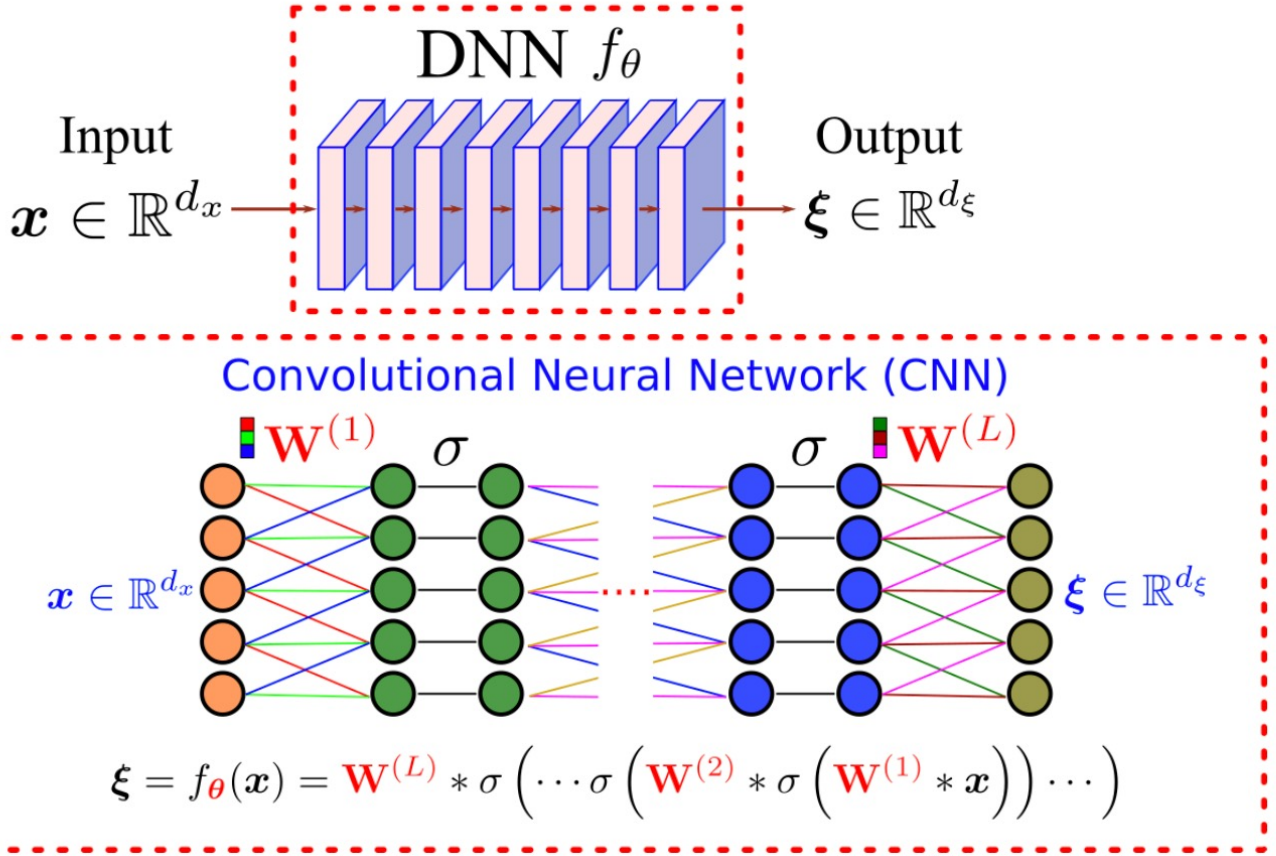
Deep Neural Network

$f_\theta : \mathbb{R}^{d_x} \to \mathbb{R}^{d_\xi}$, and θ is the collection of all trainable parameters

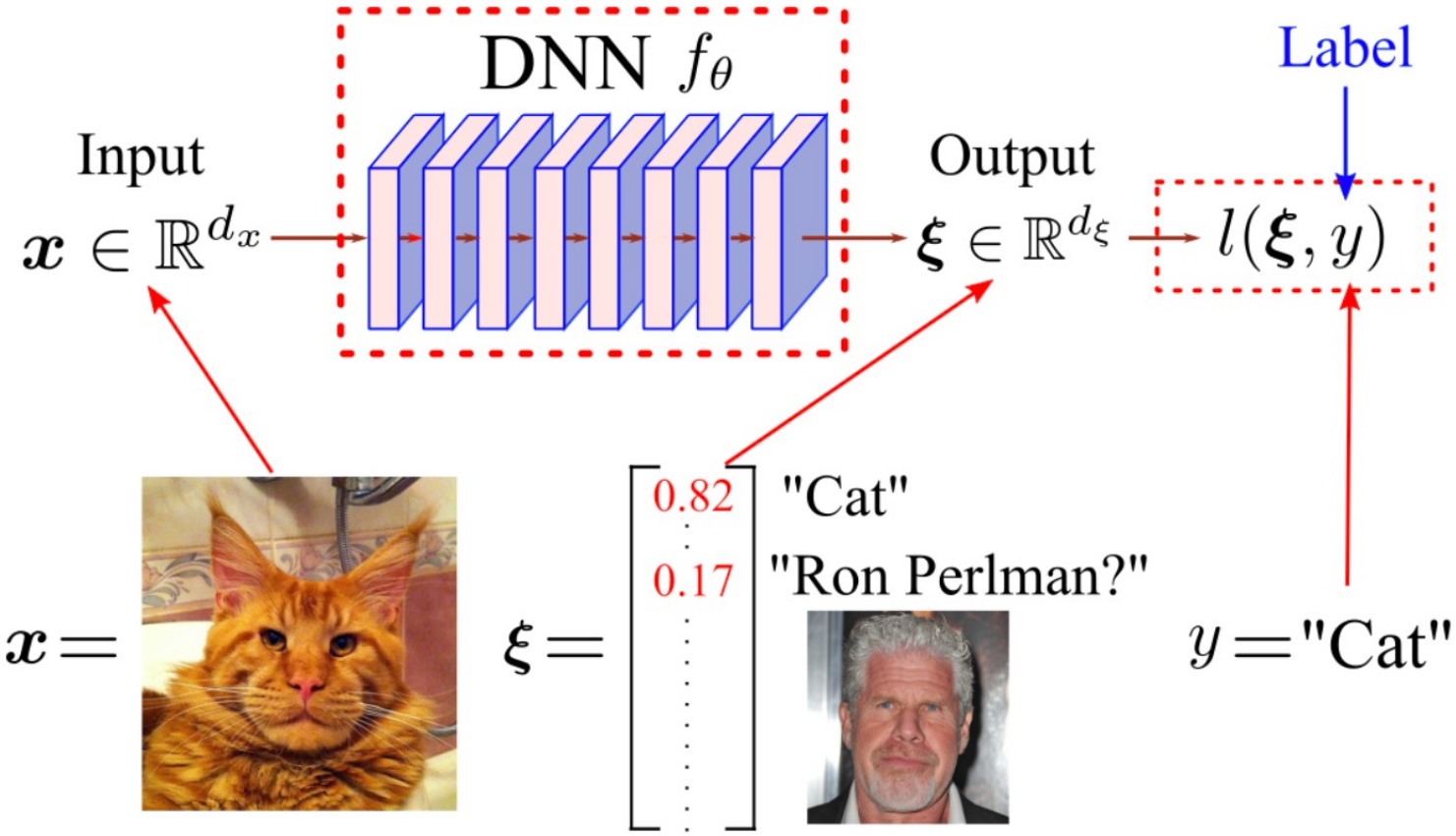


Convolutional Neural Network

$f_\theta : \mathbb{R}^{d_x} \to \mathbb{R}^{d_\xi}$, and θ is the collection of all trainable parameters

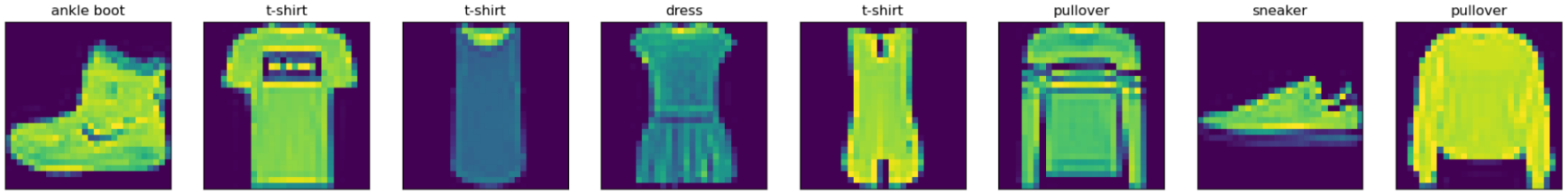


e.g. Image Classification

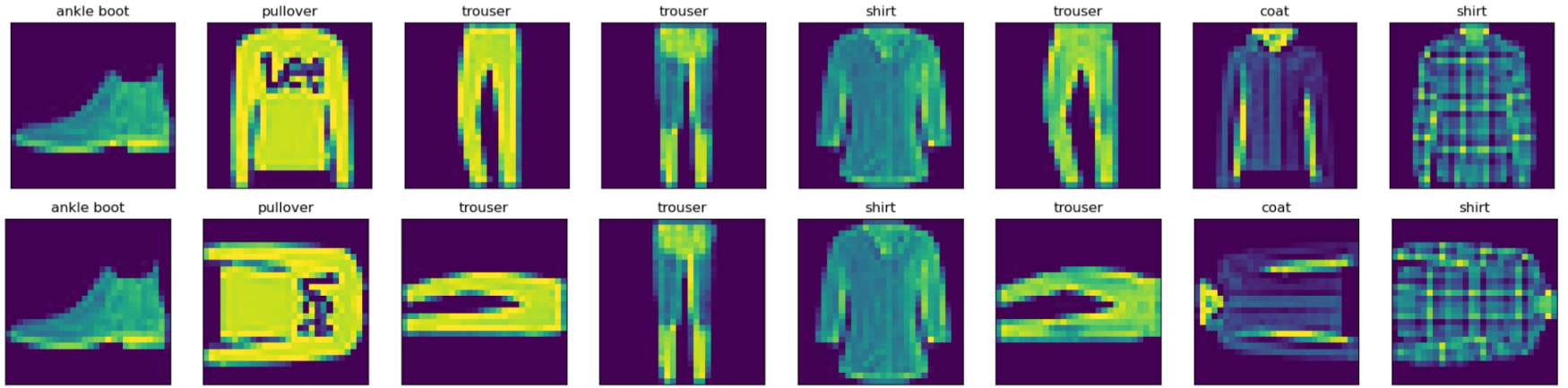


Motivating Example: Classification of Fashion-MNIST Dataset

Training Data:



Testing Data:



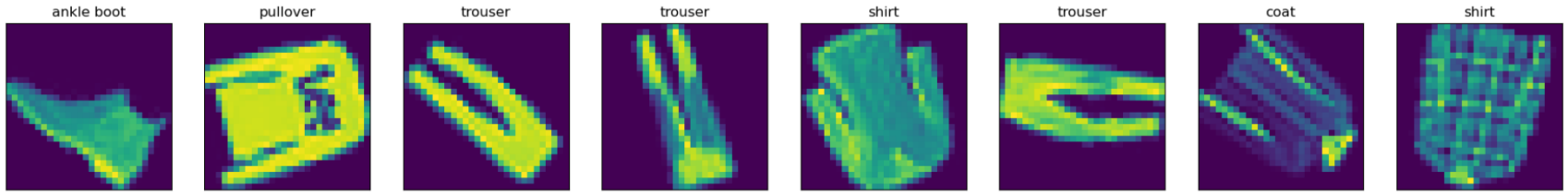
Original

Accuracy of the network: **77.9%**
Accuracy of t-shirt: 64.5%
Accuracy of trouser: 97.2%
Accuracy of pullover: 72.1%
Accuracy of dress: 78.5%
Accuracy of coat: 61.7%
Accuracy of sandal: 87.4%
Accuracy of shirt: 50.5%
Accuracy of sneaker: 89.5%
Accuracy of bag: 90.5%
Accuracy of ankle boot: 92.6%

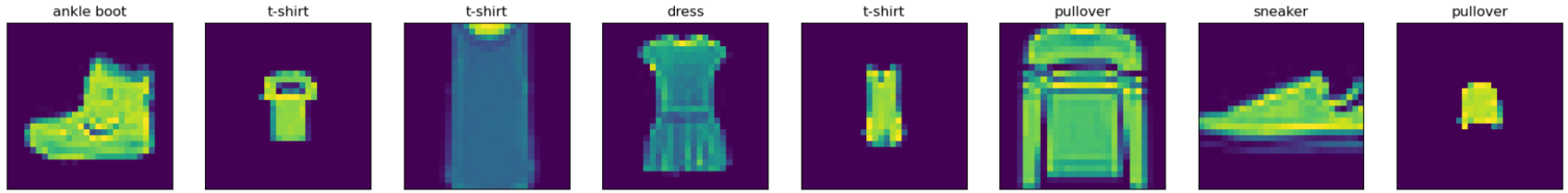
Rotated within {0, 90, 180, 270}

Accuracy of the network: **26.0%**
Accuracy of t-shirt: 22.4%
Accuracy of trouser: 35.2%
Accuracy of pullover: 19.8%
Accuracy of dress: 23.6%
Accuracy of coat: 20.0%
Accuracy of sandal: 35.6%
Accuracy of shirt: 26.8%
Accuracy of sneaker: 16.8%
Accuracy of bag: 35.8%
Accuracy of ankle boot: 26.3%

Random Rotate within [0, 360]



Random Scaled



1. Group Theory

1.1. Group

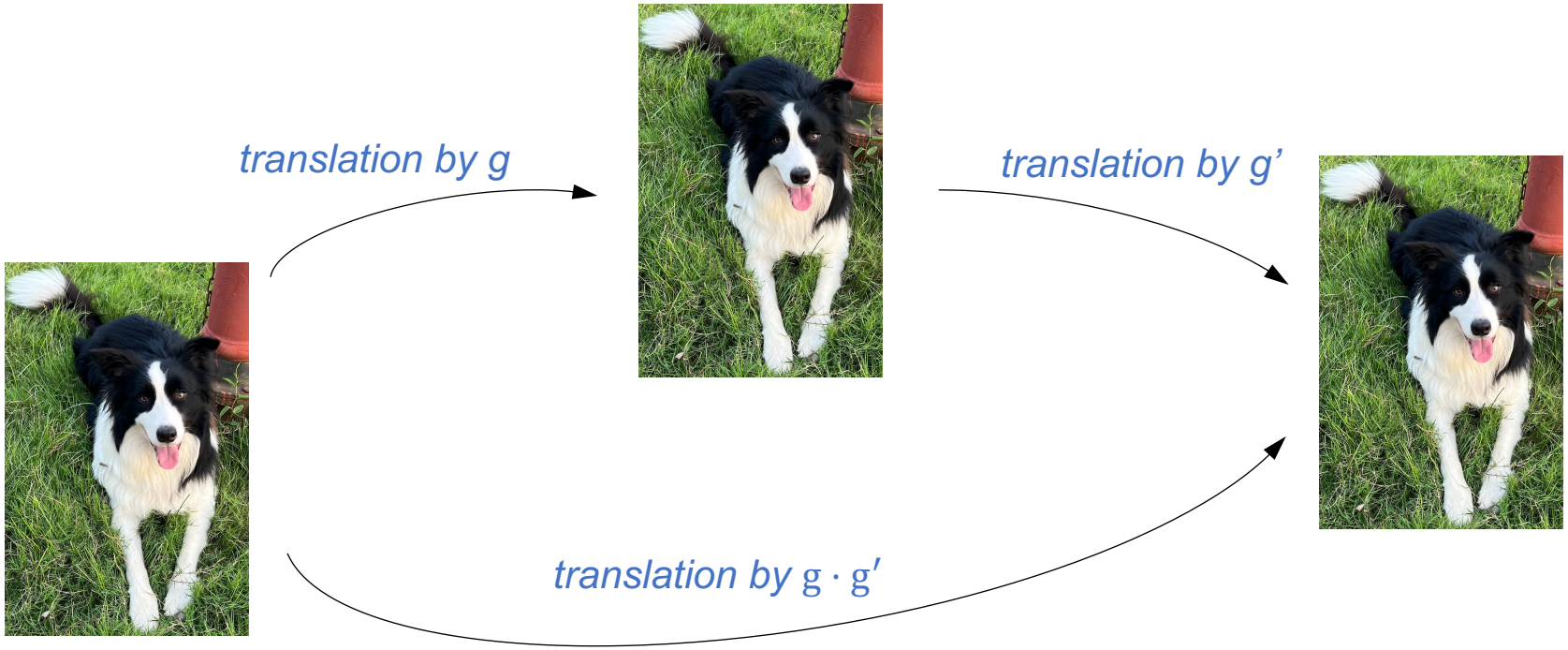
Definition of group: A group (G, •) is a set of element G equipped with a group product • , a binary operator, that satisfies the following group axioms:

- Identity: there exists an identity element $e \in G$ s.t. for any $g \in G$, $e \cdot g = g \cdot e = g$.
- Associativity: For $a, b, c \in G$, the product is associative $(a \cdot b) \cdot c = a \cdot (b \cdot c)$.
- Closure: for all a, b in G, the product $a \cdot b$ is also in G
- Inverse: for each $g \in G$, there exists an inverse element $g^{-1} \in G$ s.t. $g^{-1} \cdot g = g \cdot g^{-1} = e$

1.2 Translation Group $(\mathbb{R}^2, +)$

Consist of all translation in $\mathbb{R}^2$ and equipped with group product and group inverse:

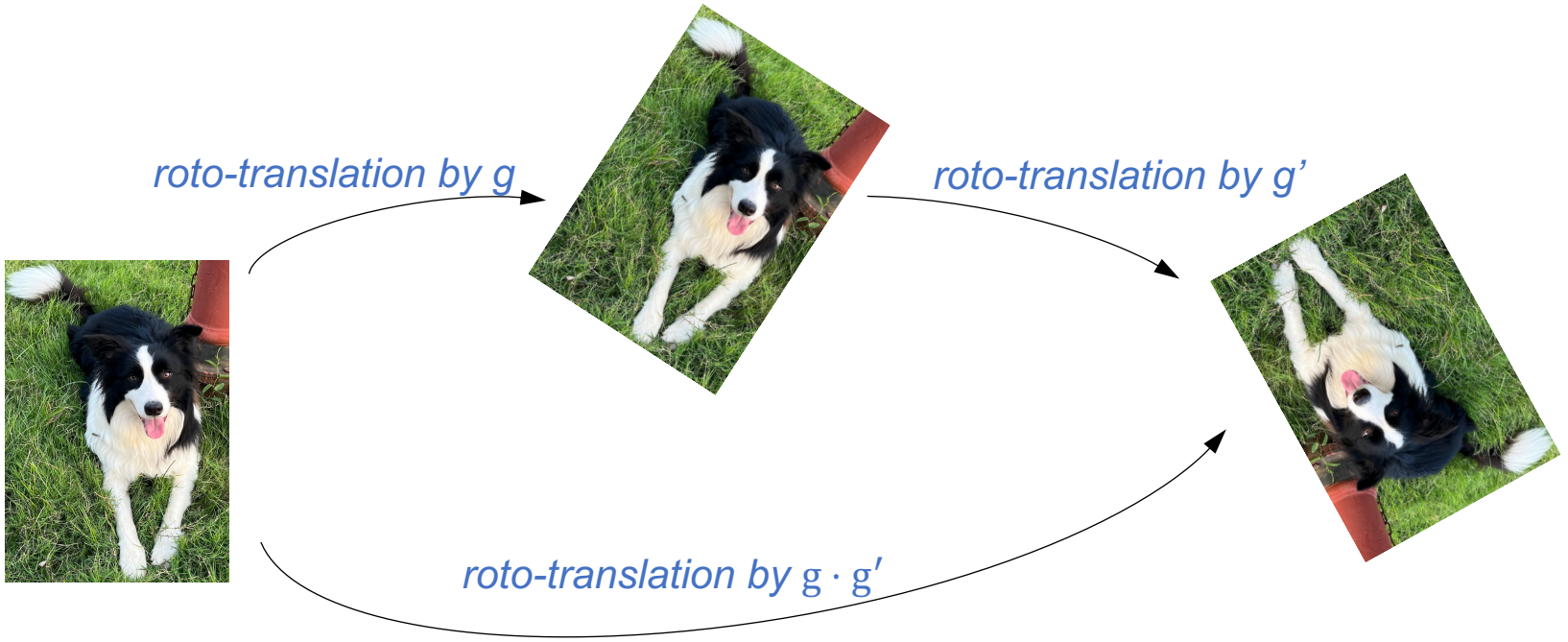
$$\mathrm{g} \cdot \mathrm{g}' = (\mathrm{x} + \mathrm{x}')$$
$$\mathrm{g}^{-1} = (-x)$$
$$with\ \mathrm{g} = (\mathrm{x}), \mathrm{g}' = (\mathrm{x}'),\ \mathrm{x}, \mathrm{x}' \in \mathbb{R}^2$$



1.3 Roto-translation Group $SE(2)$

The $SE(2) = \mathbb{R}^2 \rtimes SO(2)$ consists of the coupled space of translation vectors in $\mathbb{R}^2$ and rotation in $SO(2)$, and is equipped with group product and group inverse:

$$\mathrm{g} \cdot \mathrm{g}' = (\mathrm{R}_\theta, \mathrm{x}) \cdot (\mathrm{R}_{\theta'}, x') = (R_{\theta+\theta'}, \mathrm{R}_\theta \mathrm{x}' + \mathrm{x})$$
$$\mathrm{g}^{-1} = (\mathrm{R}_\theta^{-1}, -\mathrm{R}_\theta^{-1}\mathrm{x})$$
$$with\ \mathrm{g} = (R_\theta, \mathrm{x}), \mathrm{g}' = (R_{\theta'}, \mathrm{x}')\ , R_\theta,\ R_{\theta'} \in SO(2);\ \mathrm{x}, \mathrm{x}' \in \mathbb{R}^2$$

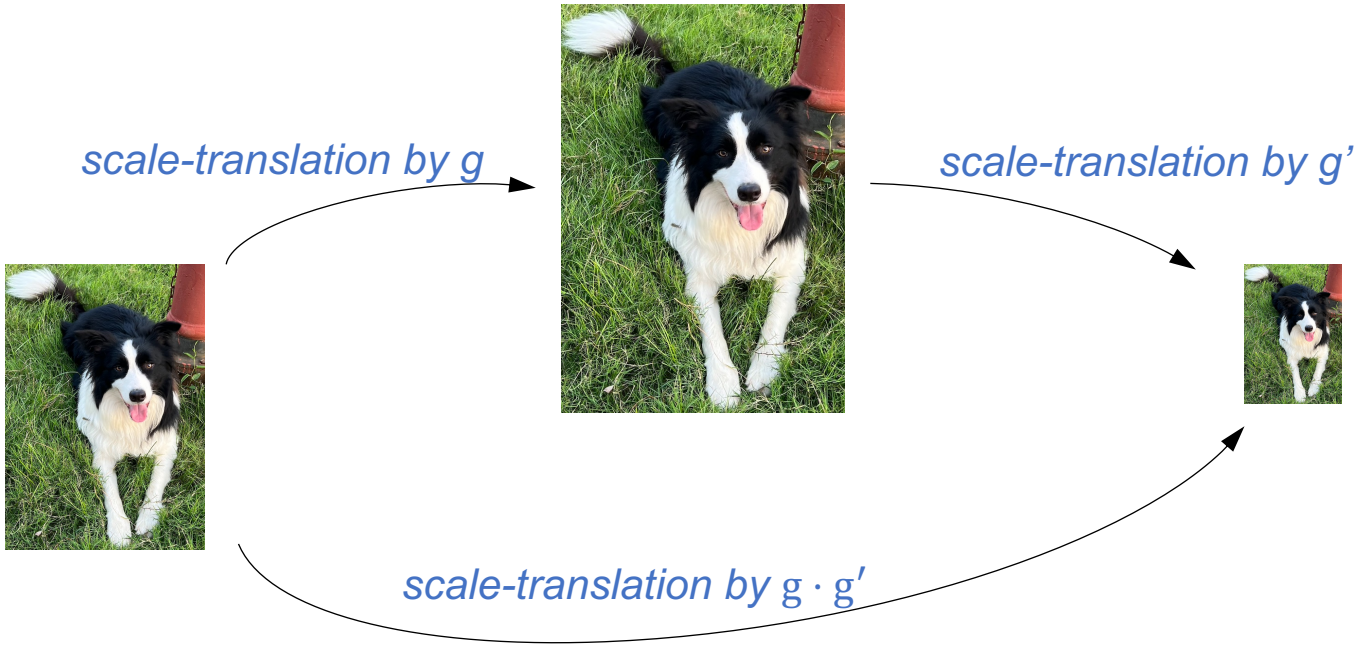


Slide credit: E. Bekkers

1.4 Scale-translation Group H

The group $\mathrm{H} = \mathbb{R}^2 \rtimes \mathbb{R}^+$ consists of the coupled space of translation vectors in $\mathbb{R}^2$ and scale/dilation factors in $\mathbb{R}^+$, and is equipped with group product and group inverse:

$$\mathrm{g} \cdot \mathrm{g}' = (s, x) \cdot (s', x') = (ss', s\mathrm{x}' + \mathrm{x})$$
$$\mathrm{g}^{-1} = (s^{-1}, -s^{-1}\mathrm{x})$$
$$\mathrm{g}'^{-1} \cdot \mathrm{g} = (s', \mathrm{x}')^{-1} \cdot (s, \mathrm{x}) = (s'^{-1}s, s'^{-1}(\mathrm{x} - \mathrm{x}'))$$
$$with\ \mathrm{g} = (s, \mathrm{x}), \mathrm{g}' = (s', \mathrm{x}'),\ \ s, s' \in \mathbb{R}^+; \mathrm{x}, \mathrm{x}' \in \mathbb{R}^2$$



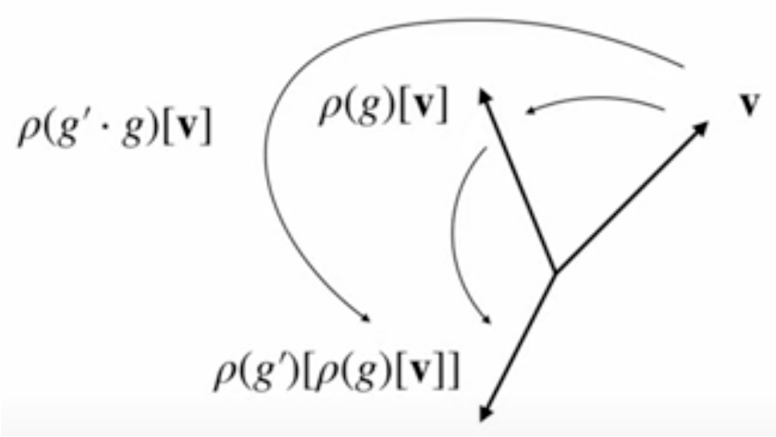
Slide credit: E. Bekkers

1.5 Group Representation

The representation $\rho: G \rightarrow GL(V)$ is a group homomorphism from G to the general linear group GL(V).

$\rho(g)$ is a linear transformation that is parameterized by group elements g, g' $\in G$ that transform some vector $v \in V$ (e.g. an image) s.t. preserves the same group structure.

$$\rho(g') \circ \rho(g)[v] = \rho(g' \cdot g)[v] \quad \forall g, g' \in G$$



Slide credit: E. Bekkers

1.6 Left-regular Representations

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

$$\mathcal{L}_g\,[f](x) \coloneqq f(g^{-1} \cdot x)$$

e.g. $f \in \mathrm{L}_2(\mathbb{R}^2)$, $G = (\mathbb{R}^2, +)$, $i.e.$, The translation group, with g represents a pure translation $t = (u, v) \in \mathbb{R}^2$

$$\mathcal{L}_g\,[f](x) = f(g^{-1} \cdot x) = f(x - t)$$

e.g. $f \in \mathrm{L}_2(\mathbb{R}^2)$, $G = SE(2)$, $i.e.$, The roto−translation group, with g represents $(\mathrm{R}_\theta, \mathrm{y})$ s. t. $R_\theta \in SO(2)$, $\mathrm{y} \in \mathbb{R}^2$

$$\mathcal{L}_g\,[f](x) = f(g^{-1} \cdot x) = f\left(\left(\mathrm{R}_\theta^{-1}, -\mathrm{R}_\theta^{-1} y\right) \cdot (0, x)\right) = f(\mathrm{R}_\theta^{-1}(x - y))$$

Note: Feature maps in G-CNNs are function on the group G, the definition of $\mathcal{L}_g$ still valid if we simply replace $\mathrm{x} \in \mathbb{Z}^2$ by $h \in G$.

$$\mathcal{L}_g[f](h) \coloneqq f(g^{-1} \cdot h)$$

1.7 Group Actions

Group product: (The action on G)

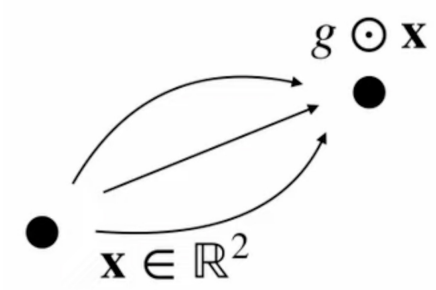
$$g \cdot g'$$

Left regular representation: (The action on $\mathrm{L}_2(\mathbb{R}^2)$)

$$\mathcal{L}_g f$$

Group action: (The action on $\mathbb{R}^d$)

$$g \odot x$$



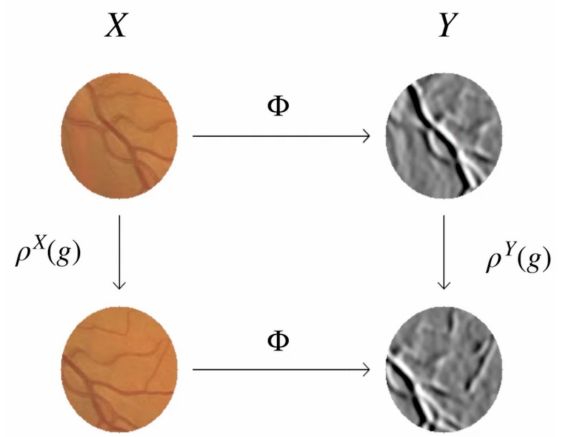
Slide credit: E. Bekkers

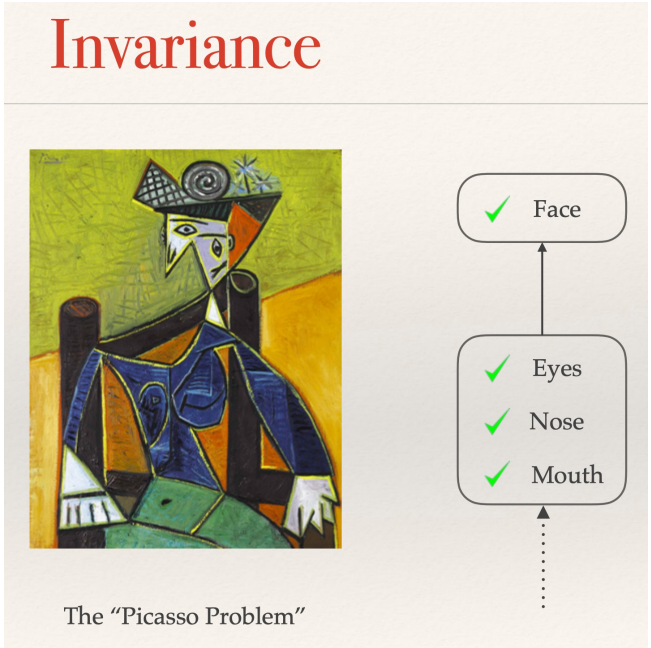
2. Group Equivariant Convolution Neural Networks

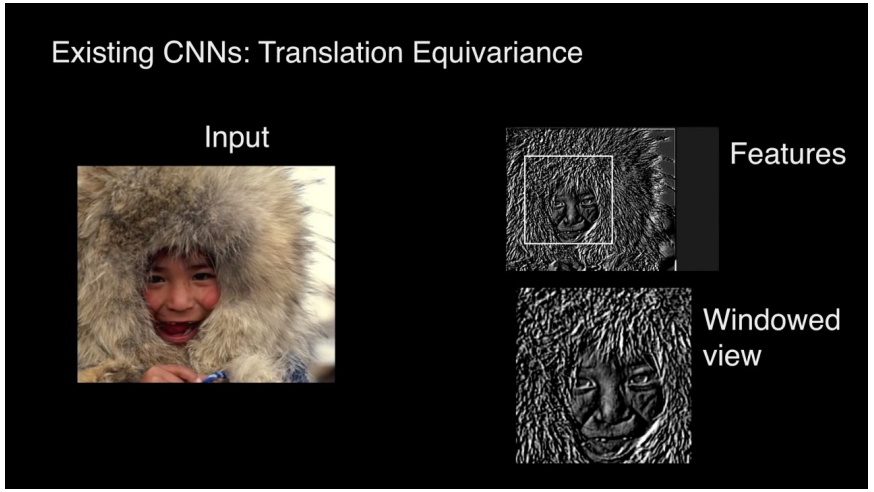
2.1 Invariance and Equivariance

Define a network or layer $\Phi: \mathrm{X} \rightarrow \mathrm{Y}$, Φ is G-equivariant if it commutes the group action:

$$\Phi \circ \rho_X(g)x = \rho_Y(g) \circ \Phi(x), \qquad \forall g \in \mathrm{G}$$

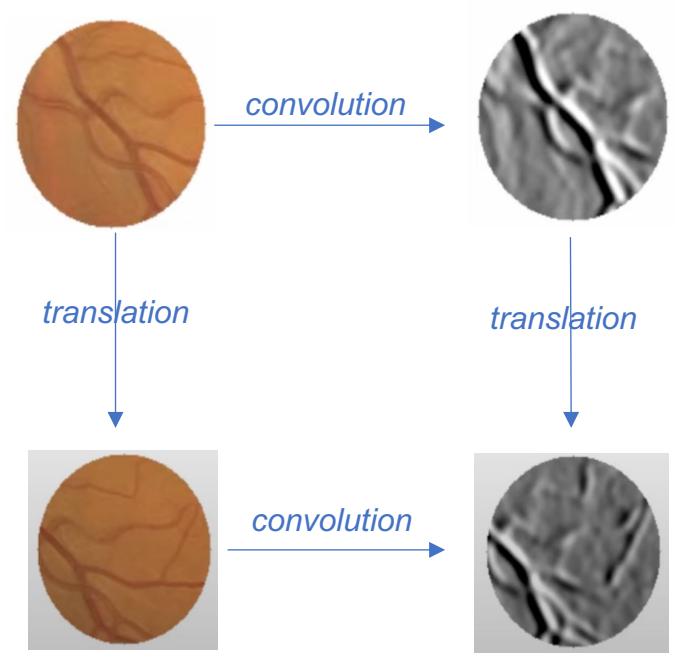




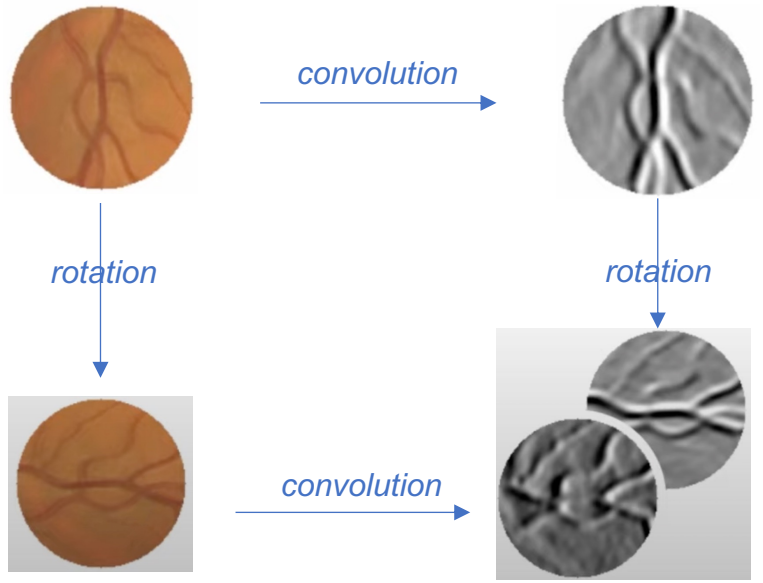


Slide credit: E. Bekkers

2.2 CNN is translation equivariant but not roto-translation equivariant



$$
\begin{aligned}
[[L_t f] \star \psi](x) &= \sum_y f(y-t)\psi(y-x) \\
&= \sum_y f(y)\psi(y+t-x) \\
&= \sum_y f(y)\psi(y-(x-t)) \\
&= [L_t[f \star \psi]](x).
\end{aligned}
$$



Rotation and convolution are not commute,
That is the problem we want to solve.

$$[[L_r f] \star \psi](x) = L_r[f \star [L_{r^{-1}}\psi]](x)$$

Slide credit: E. Bekkers

2.3 2-D convolution

Define input image/feature map $f \in \mathrm{L}_2(\mathbb{R}^2)$, 2D convolution kernel $\Psi \in \mathrm{L}_2(\mathbb{R}^2)$ and the translation group $(\mathbb{R}^2, +)$. The convolution is defined as:

$$(f * \Psi)(\mathrm{x}) = \int f(x')\Psi(x' - x)dx' = \langle f, \mathcal{L}_{(\mathbb{R}^2,+)}(x)\Psi \rangle$$

So 2D convolution can be simplified as the inner product of input f and the translated filter.

$e.g.\ f \in \mathrm{L}_2(\mathbb{R}^2),\ G = (\mathbb{R}^2, +),\ i.e.,$ The translation group,
with g represents a pure translation $t = (u, v) \in \mathbb{R}^2$

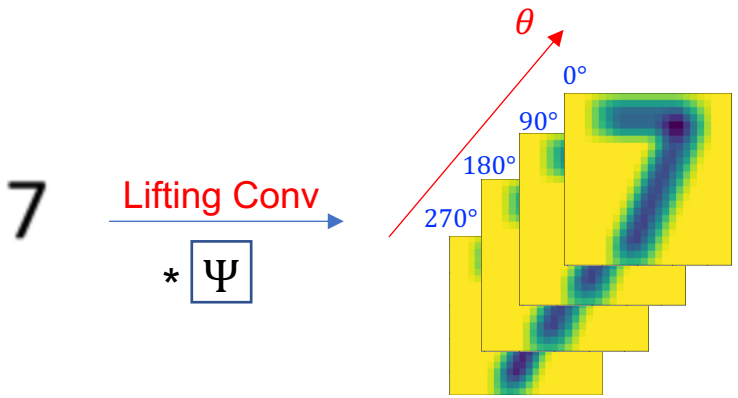
$$\mathcal{L}_g\,[f](x) = f(g^{-1} \cdot x) = f(x - t)$$

2.4 G-CNNs

2.4.1 Layer 1: SE(2) Lifting Convolution

Define $\mathrm{G} = SE(2) = \mathbb{R}^2 \rtimes SO(2)$, input image $f \in \mathrm{L}_2(\mathbb{R}^2)$, 2D convolution kernel $\Psi \in \mathrm{L}_2(\mathbb{R}^2)$. The convolution is defined as:

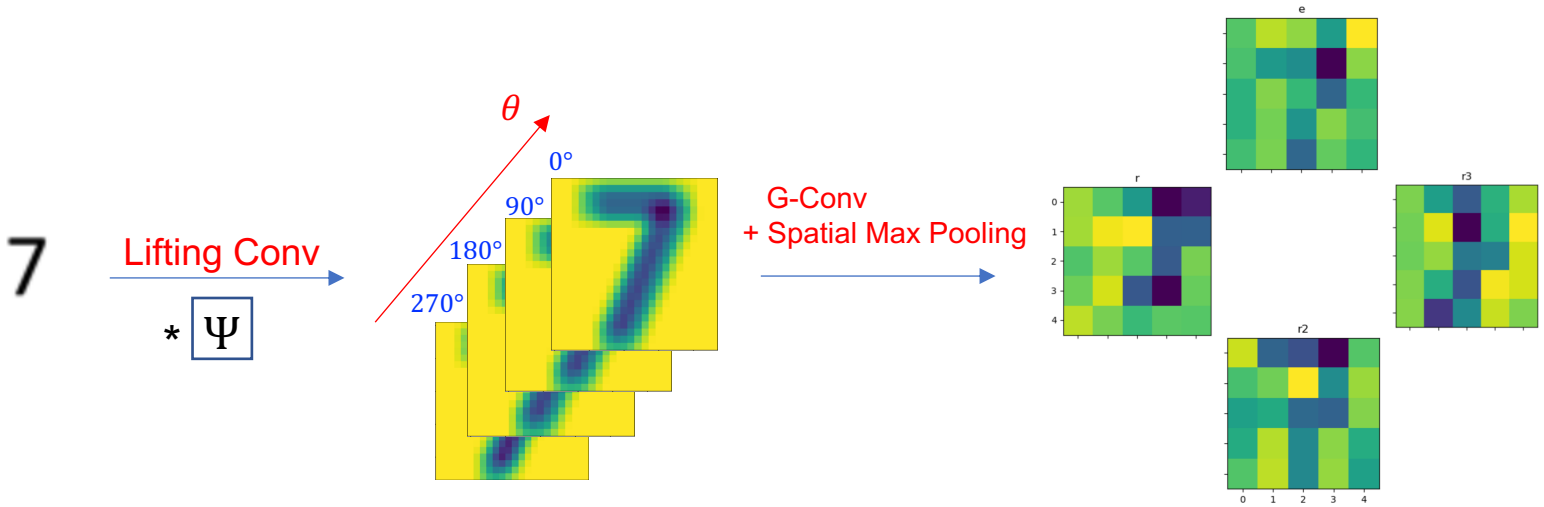
$$(f \hat{*} \Psi)(\mathrm{x}, \theta) = \int f(x')\Psi(R_\theta^{-1}(x' - x))dx' = \langle f, \mathcal{L}_{SE(2)}(x, \theta)\Psi \rangle$$



2.4.2 Layer 2: SE(2) Group Convolution

Define $\mathrm{G} = SE(2) = \mathbb{R}^2 \rtimes SO(2)$, input 3D feature map $f_g \in \mathrm{L}_2(SE(2))$, 3D convolution kernel $\Psi_g \in \mathrm{L}_2(SE(2))$. The convolution is defined as:

$$\left(f_g \hat{*} \Psi_g\right)(\mathrm{x}, \theta) = \int_{\mathbb{R}^2} \int_{SO(2)} f_g(x', \theta') \Psi_g\left(R_\theta^{-1}(x' - x), \theta - \theta'\right) d\theta dx = \left\langle f_g, \mathcal{L}_{SE(2)}(x, \theta) \Psi_g \right\rangle$$



Proof of equivariance: by using substitution $h \to \mu h$

$$
\begin{aligned}
[[L_u f] \star \psi](g) &= \sum_{h \in G} \sum_{k} f_k(u^{-1}h)\psi(g^{-1}h) \\
&= \sum_{h \in G} \sum_{k} f(h)\psi(g^{-1}uh) \\
&= \sum_{h \in G} \sum_{k} f(h)\psi((u^{-1}g)^{-1}h) \\
&= [L_u[f \star \psi]](g)
\end{aligned}
$$

2.4.3 Pointwise Non-linearities

Define the nonlinearity mapping $v: \mathbb{R} \to \mathbb{R}$, composition operator C_v. Apply v to a feature map amounts to function composition.

$$C_v\ f(g) = [v \circ f](g) = v(f(g))$$

$$C_v\ L_h f = v \circ (f \circ h^{-1}) = [v \circ f] \circ h^{-1} = L_h C_v f$$

So the rectified feature map inherits the transformation properties of the previous layer, which is also equivariant.

2.4.4 Subgroup Pooling and Coset Pooling

Define the pooling operator P applied to a feature map $\mathrm{f}: G \to \mathbb{R}$.

$$Pf(g) = \max_{k \in gH} f(k) \ \text{s.t.}\ gH = \{gh | h \in H\}, \text{left cosets of H in G}, g \in G, H \subset G$$

$$PL_h f = L_h P f$$

Pooling commutes with group action L_h. We can obtain full G-equivariance by choosing H which is a subgroup of G.

Input: [1, 3, 28, 28]

Lifting Conv.

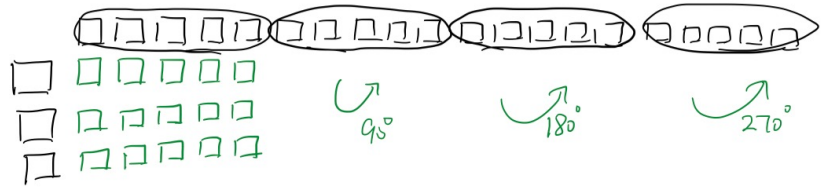
with kernel: [5, 3, 7, 7]

# of output channel

# of input channel

[1, 20, 28, 28] = [1, 4x5, 28, 28]

spatial dim



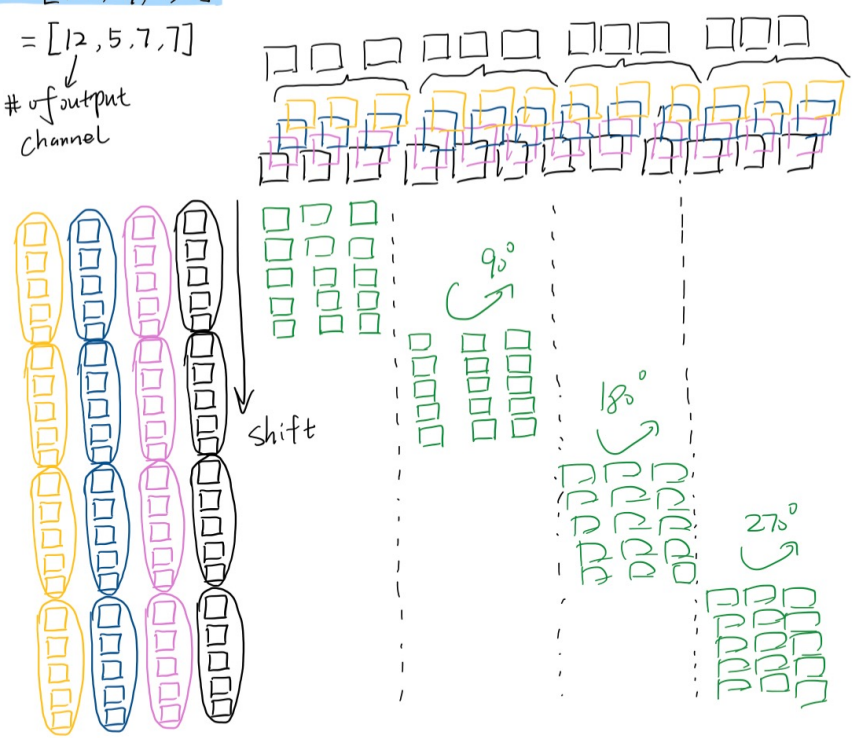
Group Conv.

with kernel: [4x3, $\frac{20}{4}$, 7, 7]

= [12, 5, 7, 7]

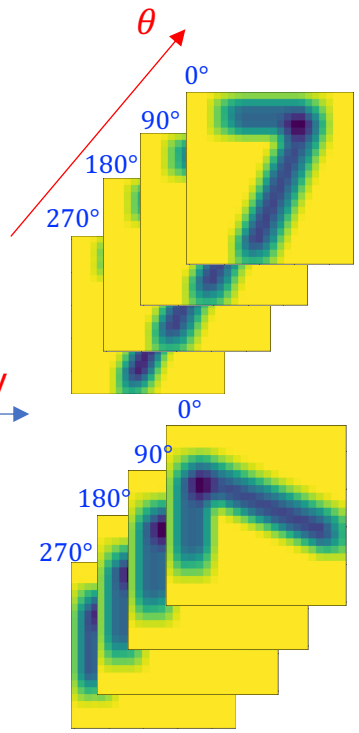
# of output channel

[1, 12, 28, 28] = [1, 4x3, 28, 28]



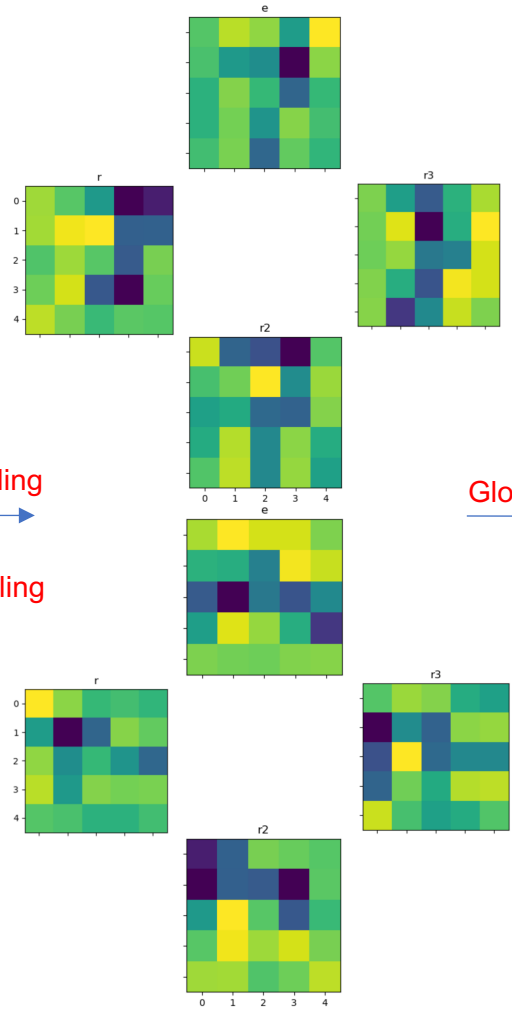
Lifting Conv

$* \boxed{\Psi}$



G-Conv1
+ Spatial Max Pooling

G-Conv2
+ Spatial Max Pooling



Global Pooling

tensor([0.5279, 0.5587, 0.5501, 0.5451]

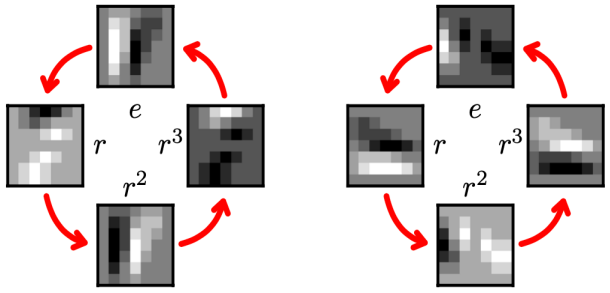
tensor([0.5279, 0.5587, 0.5501, 0.5451]

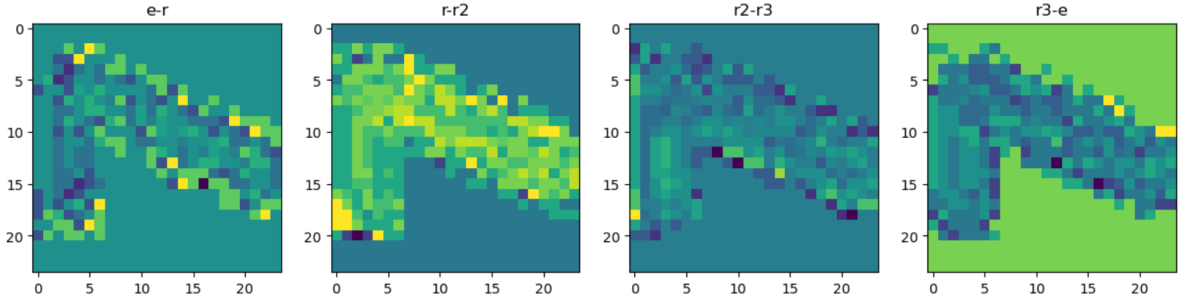
Fully Connected Layer

[-0.2988, -0.0289, -0.1721, -0.1802, 0.3358, 0.0529, 0.1850, -0.1459, 0.3403, 0.0621],

[-0.2988, -0.0289, -0.1721, -0.1802, 0.3358, 0.0529, 0.1850, -0.1459, 0.3403, 0.0621]

Figure 1. A p4 feature map and its rotation by r.





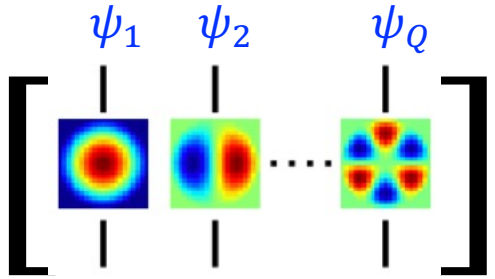
min = -7.6294e-06, max = 5.7220e-06

Q: What if rotated by multiple of 45° OR any angle other than 90°?

3. Rotation-equivariant Steerable Filter CNNs

3.1 Steerable filter

A filter is rotationally steerable when its **rotation by an arbitrary angle $\boldsymbol{\theta}$** can be expressed in a function space spanned by a fixed set of atomic basis functions $\{\Psi_q\}_{q=1}^{Q}$.



Convolution Filters: linear combinations of the elementary filters.

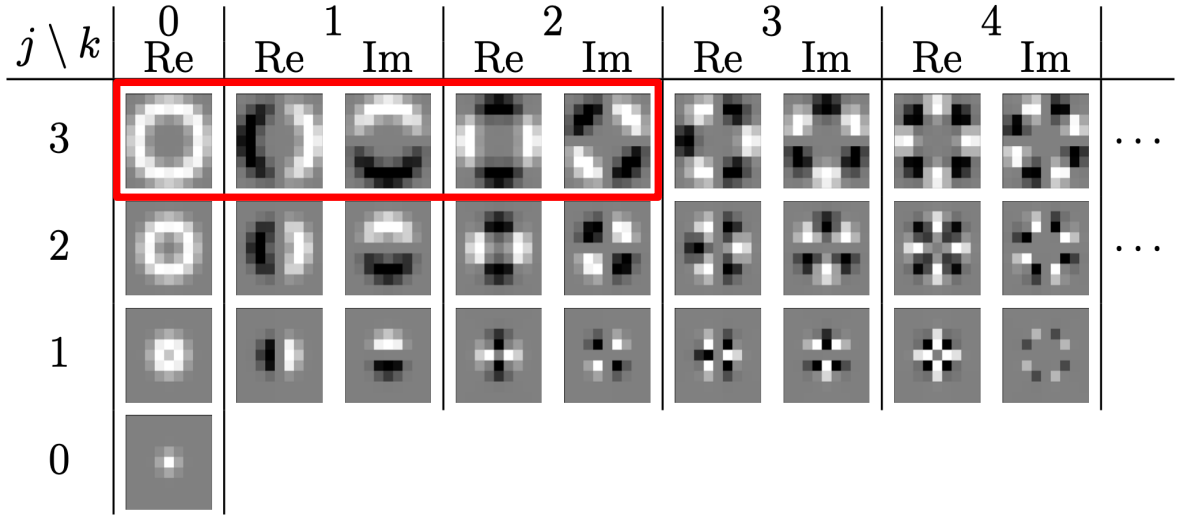
3.2 Circular Harmonics Basis:

Define sinusoidal angular part $e^{ik\Phi}$; radial function $\tau(r)$: $\mathbb{R}^+ \rightarrow \mathbb{R}$.
The circular harmonics basis is defined as:

$$\Psi_{jk}(r,\Phi) = \tau_j(r)e^{ik\Phi} = \tau_j(r)[\cos(k\Phi) + i \cdot sin(k\Phi)]$$

e.g. Choose Gaussian radial part:

$$\tau_j(r) = \exp\left(-(\mathrm{r} - \mu_j)^2 \backslash 2\sigma^2\right), \mu_j = j$$



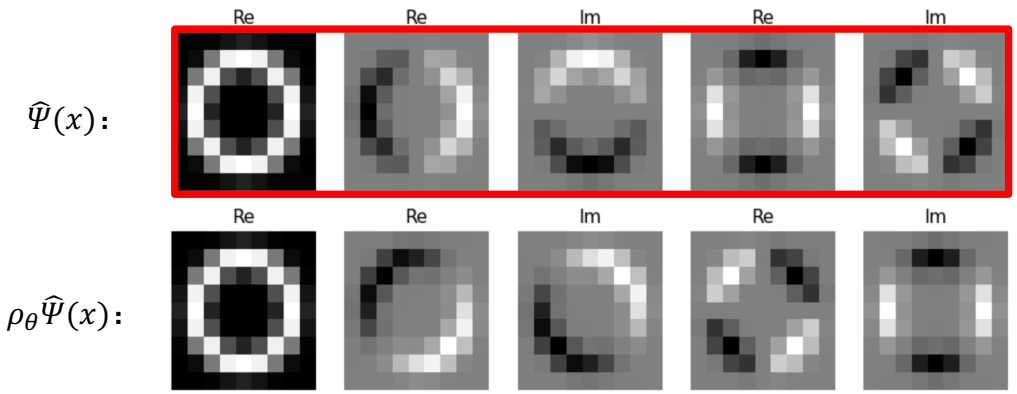
Define angular expansion coefficient functions $k_q(\theta)$, and rotation operator ρ_θ. The steerable filter $\Psi: \mathbb{R}^2 \to \mathbb{R}$ satisfies:

$$\rho_\theta \Psi(x) = \Psi(\rho_{-\theta} x) = \sum_{q=1}^{Q} k_q(\theta) \Psi_q(x), \theta \in (-\pi, \pi]$$

The composed filter can be steered as a whole (by phase manipulation of the atoms):

e.g. Circular Harmonics Basis

$$\rho_\theta \widehat{\Psi}(x) = \sum_{j=1}^{J} \sum_{k=0}^{K_j} w_{jk} e^{-ik\theta} \Psi_{jk}(x)$$



Pros of Steerability:

- Does not suffer from interpolation artifacts compared to rotations by interpolation.
- In practice, response of each orientation can be synthesized from the atomic responses $f * \Psi_q$:

$$(f * \rho_\theta \Psi)(x) = \sum_{q=1}^{Q} k_q(\theta)(f * \Psi_q)(x)\,, \theta \in (-\pi, \pi]$$

Input: [1, 3, 28, 28]

Lifting Conv.

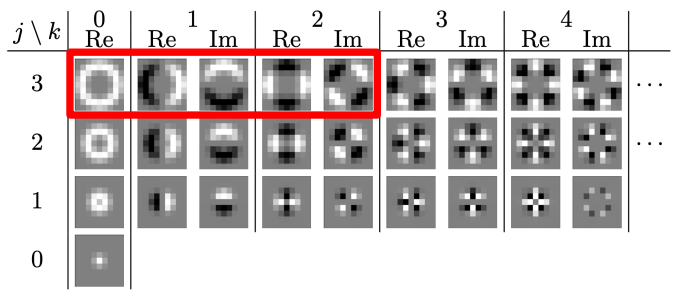
with kernel: [5, 3, 7, 7]

# of output channel

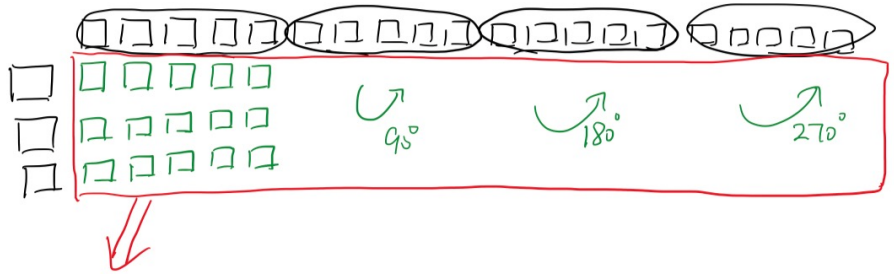
# of input channel

[1, 20, 28, 28] = [1, 4×5, 28, 28]

spatial dim



Given: Steerable Filter Basis



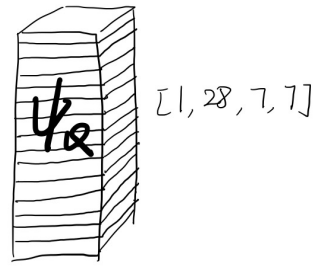
ψ = [6, 3, 5, 7, 7]

Step 1: Initialize W:

[3×5, 28]

# of input

# of output



[1, 28, 7, 7]

$G\psi_\alpha$: [6, 28, 7, 7]

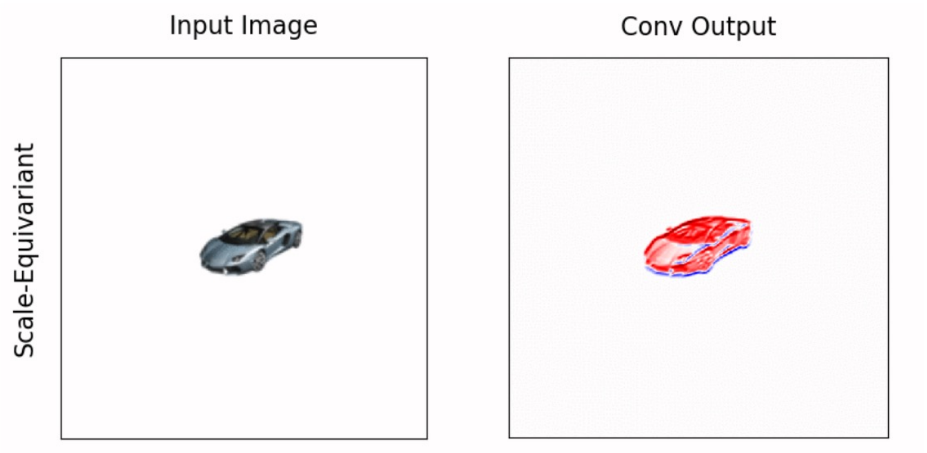
Step 2: Filter Assembling:

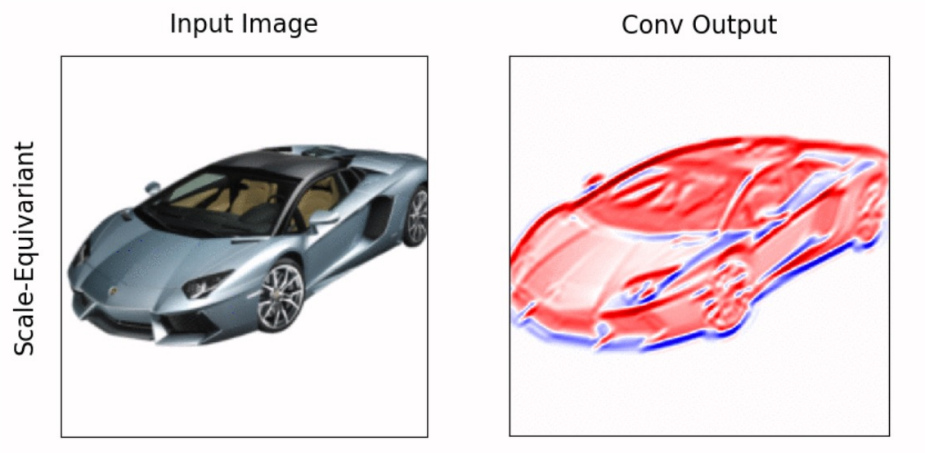
$G\psi_\alpha \oplus W$ = [6, 28, 7, 7] ⊕ [3×5, 28]

[6, 3×5, 7, 7]

Q: What if we want a network:
invariant to scale change + translation equivariant?

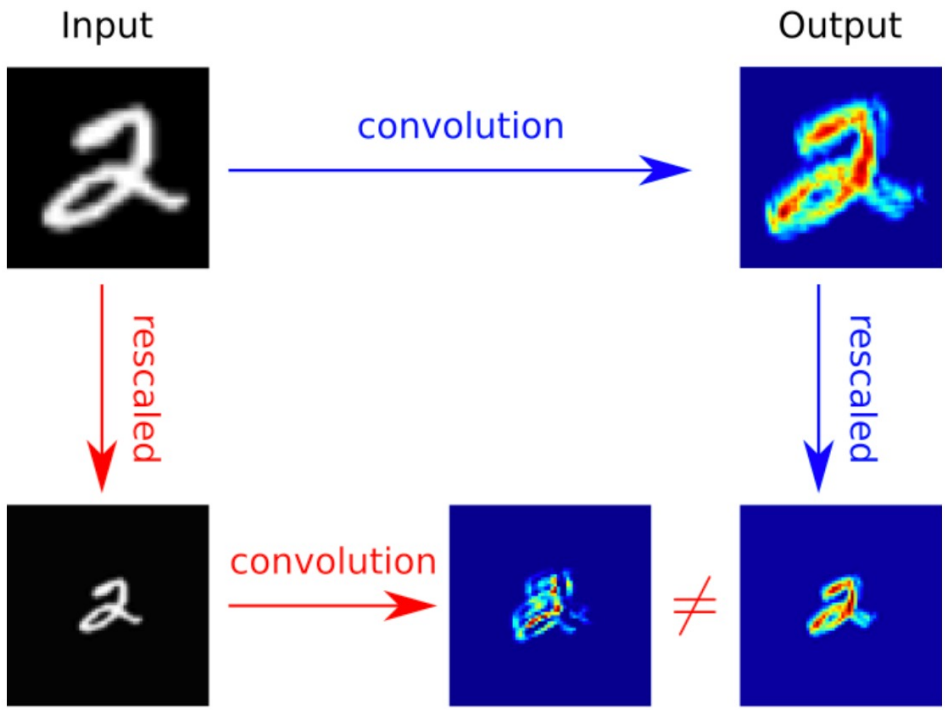
e.g. image segmentation





Slide credit: I. Sosnovik, M. Szmaja, A Smeulders

CNNs are NOT Scale-Equivariant



4. Scale-Equivariant Steerable Filter CNNs

4.1 Scale-Equivariant Mapping:

Given the function $f(s,t)$ and a steerable filter $\Psi_\sigma(s,t)$ defined on $G = \mathbb{R}^2 \rtimes \mathbb{R}^+$. A scale convolution is given by:

$$
\begin{aligned}
[f *_H \Psi_\sigma](s,t) &= \int_G f(s',t') L_{st}[\Psi_\sigma](s',t') d\,\mu(g) \\
&= \sum_{s'} \int_T f(s',t') \Psi_{s\sigma}(s^{-1}s', t'-t) dt' \\
&= \sum_{s'} [f(s',\cdot) * \Psi_{s\sigma}(s^{-1}s',\cdot)](t)
\end{aligned}
$$

4.2 Proof of Equivariant to Translation:

$$
\begin{aligned}
[L_{\hat{t}}[f] \star_H \psi_\sigma](s,t) &= \sum_{s'} [L_{\hat{t}}[f](s',\cdot) \star \psi_{s\sigma}(s^{-1}s',\cdot)](t) \\
&= \sum_{s'} L_{\hat{t}}[f(s',\cdot) \star \psi_{s\sigma}(s^{-1}s',\cdot)](t) \\
&= L_{\hat{t}}\Big\{ \sum_{s'} [f(s',\cdot) \star \psi_{s\sigma}(s^{-1}s',\cdot)] \Big\}(t) \\
&= L_{\hat{t}}[f \star_H \psi_\sigma](s,t)
\end{aligned}
$$

4.3 Proof of Equivariant to Scale Transformation:

$$
\begin{aligned}
[L_{\hat{s}}[f] \star_H \psi_\sigma](s,t) &= \sum_{s'} [L_{\hat{s}}[f](s',\cdot) \star \psi_{s\sigma}(s^{-1}s',\cdot)](t) \\
&= \sum_{s'} L_{\hat{s}}[f(\hat{s}^{-1}s',\cdot) \star \psi_{\hat{s}^{-1}s\sigma}(s^{-1}s',\cdot)](t) \\
&= \sum_{s''} [f(s'',\cdot) \star \psi_{\hat{s}^{-1}s\sigma}(\hat{s}s^{-1}s'',\cdot)](\hat{s}^{-1}t) \\
&= [f \star_H \psi_\sigma](\hat{s}^{-1}s, \hat{s}^{-1}t) \\
&= L_{\hat{s}}[f \star_H \psi_\sigma](s,t)
\end{aligned}
$$

4.4 Proof of Equivariance of Convolution to Transformation from $G = \mathbb{R}^2 \rtimes \mathbb{R}^+$:

By the property of semidirect product of groups:

$$L_{\hat{s}\hat{t}}[f] \star_H \psi_\sigma = L_{\hat{s}} L_{\hat{t}}[f] \star_H \psi_\sigma = L_{\hat{s}}[L_{\hat{t}}[f] \star_H \psi_\sigma] = L_{\hat{s}} L_{\hat{t}}[f \star_H \psi_\sigma] = L_{\hat{s}\hat{t}}[f \star_H \psi_\sigma]$$

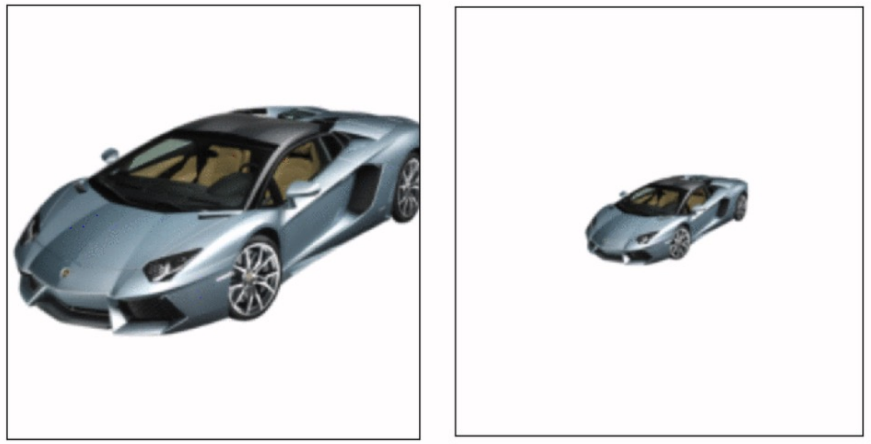
4.5 Basis:

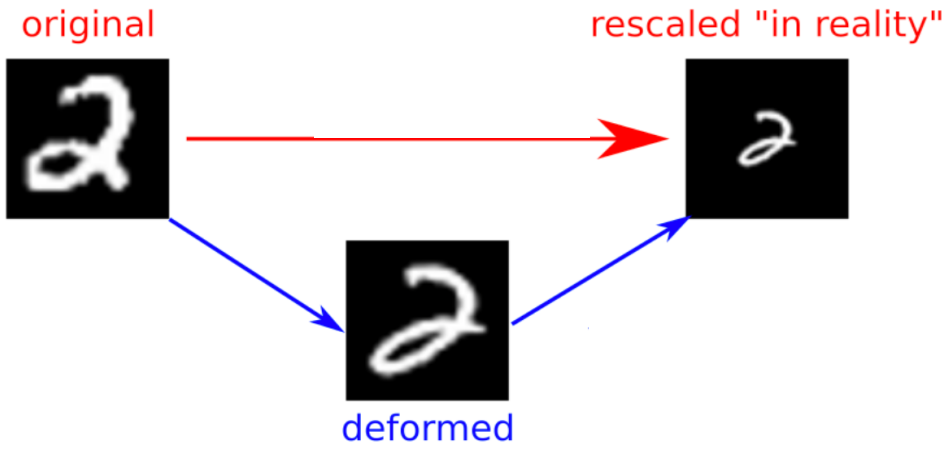
Define A is a constant independent on σ, H_n----Hermite polynomial of the n-th order.

$$\psi_\sigma(x, y) = A \frac{1}{\sigma^2} H_n\left(\frac{x}{\sigma}\right) H_m\left(\frac{y}{\sigma}\right) \exp\left[-\frac{x^2 + y^2}{2\sigma^2}\right]$$

We iterate over increasing pairs of n, m to generate required number of functions.

Q: Scaling effect in reality is not always exact. What is the robustness of equivariance to input deformation?

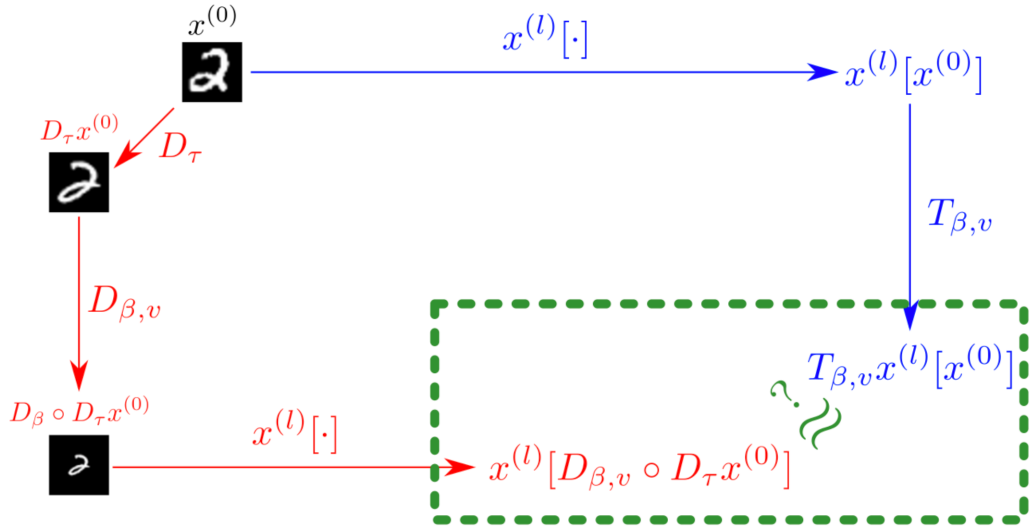


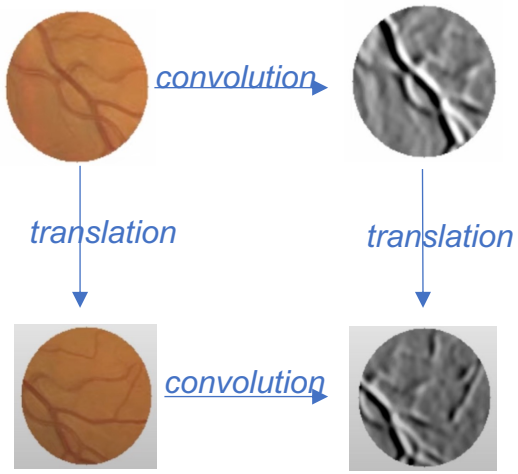


5. Stability of equivariant representation to deformation

Under some mild assumptions, define D_τ be a nonlinear spatial deformation, let $D_{\beta,v}, T_{\beta,v}$ be the 'perfect' arbitrary scaling centered at $v \in \mathbb{R}_+$. We have, for any L,

$$\left\| x^{(L)}[D_{\beta,v} \circ D_\tau x^{(0)}] - T_{\beta,v} x^{(L)}[x^{(0)}] \right\| \leq 2^{\beta+1} \left(4L|\nabla\tau|_\infty + 2^{-j_L}|\tau|_\infty\right) \|x^{(0)}\|$$





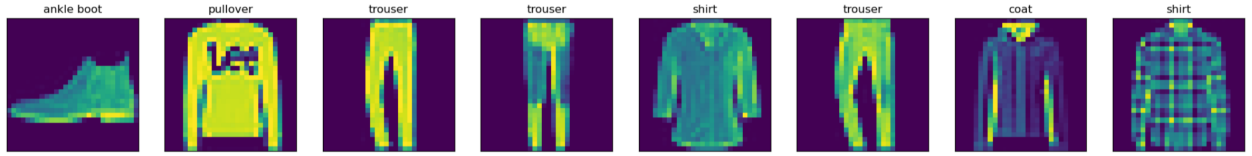
Improvement:

Zhu, W and Qiu, Q and Calderbank, R and Sapiro, G and Cheng, X. Scaling-Translation-Equivariant Networks with Decomposed Convolutional Filters, JMLR 2022.

6. Experiment on Fashion-MNIST Dataset

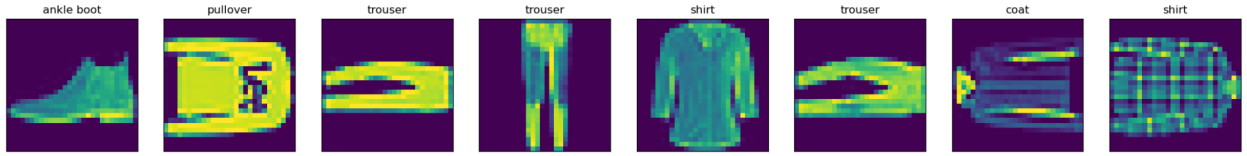
6.1 G-CNNs v.s. CNNs:

Training Data:



# of epochs: 50
Batch size: 64
Learning rate: 0.01
Optimizer: Adam with decaying rate 1e-4

Testing Data:



CNNs: # of parameters: 9650

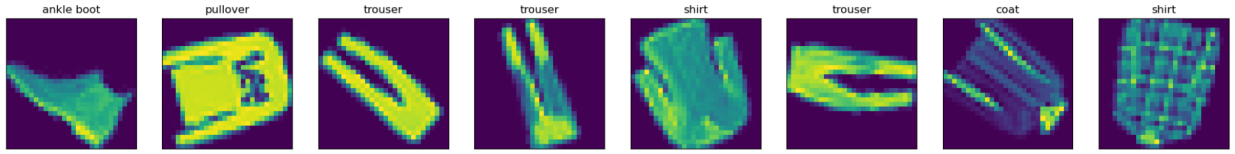
Accuracy of the network: **26.0%**
Accuracy of t-shirt: 22.4%
Accuracy of trouser: 35.2%
Accuracy of pullover: 19.8%
Accuracy of dress: 23.6%
Accuracy of coat: 20.0%
Accuracy of sandal: 35.6%
Accuracy of shirt: 26.8%
Accuracy of sneaker: 16.8%
Accuracy of bag: 35.7%
Accuracy of ankle boot: 26.3%

G-CNNs: # of parameters: 6755

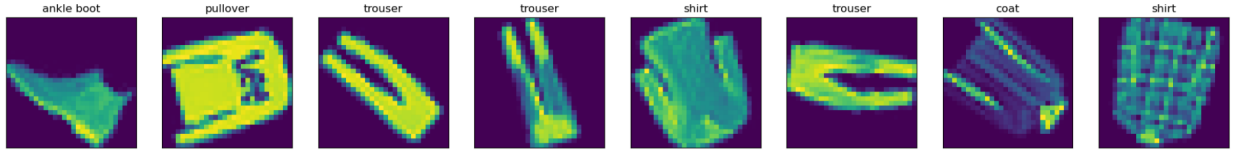
Accuracy of the network: **77.1%**
Accuracy of t-shirt: 74.6%
Accuracy of trouser: 94.3%
Accuracy of pullover: 50.9%
Accuracy of dress: 76.2%
Accuracy of coat: 64.0%
Accuracy of sandal: 86.1%
Accuracy of shirt: 61.2%
Accuracy of sneaker: 93.3%
Accuracy of bag: 77.1%
Accuracy of ankle boot: 93.2%

6.2 R-SFCNNs v.s. CNNs:

Training Data:



Testing Data:



# of epochs: 50
Batch size: 64
Learning rate: 0.01
Optimizer: Adam with decaying rate 1e-4

R-SFCNNs: # of parameters: 10939

Accuracy of the network: **74.5%**
Accuracy of t-shirt: 78.00%
Accuracy of trouser: 94.6%
Accuracy of pullover: 59.8%
Accuracy of dress: 84.7%
Accuracy of coat: 48.8%
Accuracy of sandal: 86.2%
Accuracy of shirt: 39.6%
Accuracy of sneaker: 92.0%
Accuracy of bag: 76.8%
Accuracy of ankle boot: 87.2%

CNNs: # of parameters: 11650

Accuracy of the network: **60.6%**
Accuracy of t-shirt: 57.0%
Accuracy of trouser: 79.1%
Accuracy of pullover: 67.6%
Accuracy of dress: 65.6%
Accuracy of coat: 52.2%
Accuracy of sandal: 82.8%
Accuracy of shirt: 4.12%
Accuracy of sneaker: 87.4%
Accuracy of bag: 37.9%
Accuracy of ankle boot: 74.7%

7. Future Direction

- Sample Complexity Analysis
- Implicit Bias
- Equivariant Model in Scientific Computing

e.g. fluid mechanics ---turbulence closure model