General E(2) - Equivariant Steerable CNNs

Short review of the work:

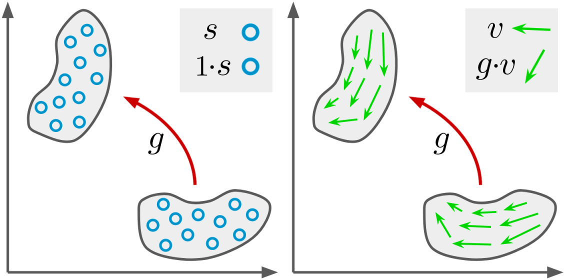
Weiler, M. and Cesa, G., 2019. General e (2)-equivariant steerable cnns. Advances in Neural Information Processing Systems, 32; ArXiv: 1911.08251

A.Demichev

June 2022

Part 1. General Theory and Basics of Implementation

E(2) - steerable feature fields



Induced representation:

$$f(x) \;\mapsto\; \left(\left[\mathrm{Ind}_G^{(\mathbb{R}^2,+)\rtimes G}\rho\right](tg)\cdot f\right)(x) \;:=\; \rho(g)\cdot f\left(g^{-1}(x-t)\right). \quad (1)$$

$(\mathbb{R}^2,+)\rtimes G$: G can be SO(2), the group $(\{\pm 1\}, *)$ of the reflections along a given axis, the cyclic groups C_N (discrete rotations by $\frac{2\pi}{N}$), the dihedral groups D_N (C_N + reflections) or the orthogonal group O(2) itself.

E(2) - steerable feature fields (2)

- In analogy to the feature spaces of vanilla CNNs comprising **multiple channels**, the feature spaces of steerable CNNs consist of multiple feature fields $f_i \colon \mathbb{R}^2 \to \mathbb{R}^{c_i}$, each of which is associated with its own *type* $\rho_i \colon G \to \mathrm{GL}(\mathbb{R}^{c_i})$.
- A common example for a stack of feature fields are RGB images $f \colon \mathbb{R}^2 \to \mathbb{R}^3$. Since the color channels transform independently under rotations we identify them as three independent scalar fields. The stacked field representation is thus given by the direct sum $\bigoplus_{i=1}^{3} 1 = \mathrm{id}_{3\times 3}$ of three trivial representations.
- While the input and output types of steerable CNNs are given by the learning task, **the user needs to specify** the types ρ_i of intermediate feature fields as hyperparameters, **similar to the choice of channels** for vanilla CNNs.

E(2) - steerable convolutions

As proven for Euclidean groups, the most general *equivariant linear map* between steerable feature spaces, transforming under ρ_{in} and ρ_{out}, is given by *convolutions* with *G-steerable kernels* $k : \mathbb{R}^2 \to \mathbb{R}^{c_{\text{out}} \times c_{\text{in}}}$, satisfying a kernel constraint

$$k(gx) \; = \; \rho_{\text{out}}(g) k(x) \rho_{\text{in}}(g^{-1}) \quad \forall g \in G, \; x \in \mathbb{R}^2 . \tag{2}$$

Intuitively, this constraint determines the form of the kernel in transformed coordinates gx in terms of the kernel in non-transformed coordinates x and thus its response to transformed input fields.

General solution of the kernel constraint for O(2) and subgroups

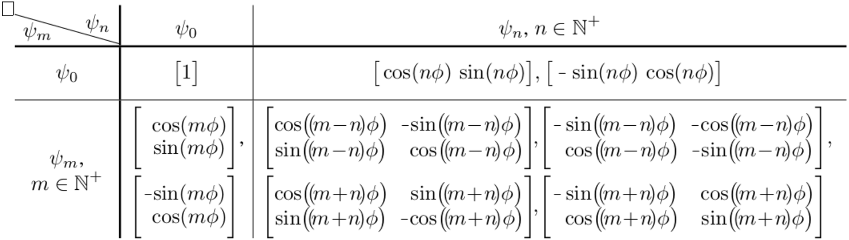
After decomposition into irreps $k(x) \to \kappa^{ij}_{\alpha\beta}$ (i, j - types + multiplicities).

Since all irreps of $O(2)$ correspond to one unique *angular frequency*, it is convenient to expand the kernel w.l.o.g. in terms of an (angular) Fourier series

$$\begin{aligned} \kappa^{ij}_{\alpha\beta}(x(r,\phi)) &= A_{\alpha\beta,0}(r) + \sum_{\mu=1}^{\infty} \Big[A_{\alpha\beta,\mu}(r)\cos(\mu\phi) \\ &\quad + B_{\alpha\beta,\mu}(r)\sin(\mu\phi) \Big] \end{aligned} \tag{3}$$

By inserting this expansion into the irrep constraints and projecting on individual harmonics we obtain constraints on the Fourier coefficients, forcing most of them to be zero.

Analytical solutions of the irrep kernel constraints. Example: $G = SO(2)$



ψ_m \ ψ_n	ψ_0	$\psi_n,\, n \in \mathbb{N}^+$
ψ_0	$\begin{bmatrix}1\end{bmatrix}$	$\begin{bmatrix}\cos(n\phi) & \sin(n\phi)\end{bmatrix}, \begin{bmatrix}-\sin(n\phi) & \cos(n\phi)\end{bmatrix}$
$\psi_m,$ $m \in \mathbb{N}^+$	$\begin{bmatrix}\cos(m\phi)\\ \sin(m\phi)\end{bmatrix},$ $\begin{bmatrix}-\sin(m\phi)\\ \cos(m\phi)\end{bmatrix}$	$\begin{bmatrix}\cos\big((m-n)\phi\big) & -\sin\big((m-n)\phi\big)\\ \sin\big((m-n)\phi\big) & \cos\big((m-n)\phi\big)\end{bmatrix}, \begin{bmatrix}-\sin\big((m-n)\phi\big) & -\cos\big((m-n)\phi\big)\\ \cos\big((m-n)\phi\big) & -\sin\big((m-n)\phi\big)\end{bmatrix},$ $\begin{bmatrix}\cos\big((m+n)\phi\big) & \sin\big((m+n)\phi\big)\\ \sin\big((m+n)\phi\big) & -\cos\big((m+n)\phi\big)\end{bmatrix}, \begin{bmatrix}-\sin\big((m+n)\phi\big) & \cos\big((m+n)\phi\big)\\ \cos\big((m+n)\phi\big) & \sin\big((m+n)\phi\big)\end{bmatrix}$

- To form the kernels such basis functions are multiplied by r-dependent function + learnable weights.
-

Group representations and nonlinearities

- A general class of representations are *unitary representations* which preserve the norm of their representation space, that is, they satisfy $|\rho_{\text{unitary}}(g) f(x)| = \left| f(x) \right| \;\; \forall\; g \in G$.
- nonlinearities which solely act on the *norm* of feature vectors but preserve their orientation are equivariant w.r.t. unitary representations $\sigma_{\text{norm}} : \mathbb{R}^c \to \mathbb{R}^c$,

  $$f(x) \mapsto \eta\big(|f(x)|\big) \frac{f(x)}{|f(x)|}$$

  for some nonlinear function $\eta : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ acting on the norm of feature vectors.
- *Norm-ReLUs*, defined by $\eta(|f(x)|) = \text{ReLU}(|f(x)| - b)$ where $b \in \mathbb{R}^{+}$ is a learned bias
- also it was considered *squashing nonlinearities* $\eta(|f(x)|) = \frac{|f(x)|^2}{|f(x)|^2 + 1}$.

Implementation details

- free to choose any radial profile, here: Gaussian radial profiles $\exp\left(\frac{1}{2\sigma^2}(r - R)^2\right)$ of width σ, centered at radii $R = 1, \ldots, \lfloor s/2 \rfloor$.

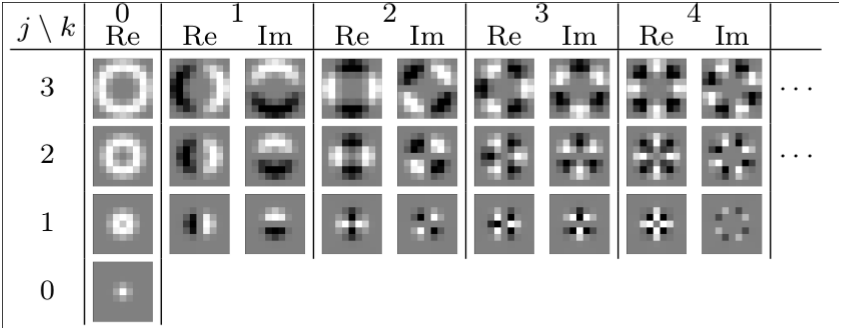


Figure 2: Illustration of the circular harmonics $\psi_{jk}(r, \phi) = \tau_j(r)\, e^{ik\phi}$ sampled on a 9×9 grid. Each row shows a different radial part j, the

Implementation details (2)

- In practice, we consider digitized signals on a pixel grid $\mathbb{Z}^2$.
  - this prevents equivariance from being exact for groups which are not symmetries of the grid (for $\mathbb{Z}^2$ - subgroups of D_4)
- sample the analytically found kernel basis $\{k_1, \ldots, k_d\}$ on a square grid of size $s \times s$ to obtain their numerical representation of shape $(d, c_{\text{out}}, c_{\text{in}}, s, s)$.
  - important to prevent aliasing effects: when the harmonics being sampled with a too low rate, a basis kernel can appear as a lower harmonic and might therefore introduce non-equivariant kernels to the sampled basis.
- radial profile: ring whose circumference $\sim$ angular sampling rate $\sim$ radius.
- It is therefore appropriate to bandlimit the kernel basis by a cutoff frequency which is chosen in proportion to the rings' radii.
- There is a work where the choice is restricted only to $m = 0, 1$ and this gives a good results
- Since the basis kernels are harmonics of specific angular frequencies this is easily implemented by discarding high

Implementation details (3)

- At runtime, the convolution kernels are expanded by contracting the sampled kernel bases with learned weights. Specifically, each basis $\left\{k_1^{\gamma\delta}, \ldots, k_{d^{\gamma\delta}}^{\gamma\delta}\right\}$, realized by a tensor of shape $(d^{\gamma\delta}, c_{\text{out},\gamma}, c_{\text{in},\delta}, s, s)$, is expanded into the corresponding block $k^{\gamma\delta}$ of the kernel by contracting it with a tensor of learned parameters of shape $(d^{\gamma\delta})$.
- This process is sped up further by batching together multiple occurrences of the same pair of representations and thus block bases.
- The resulting kernels are then used in a standard convolution routine.

Implementation details (4)

- implementation is provided as a PyTorch extension which is available at `https://github.com/QUVA-Lab/e2cnn`.
- The library provides equivariant versions of many neural network operations, including G-steerable convolutions, nonlinearities, mappings to produce invariant features, spatial pooling, batch normalization and dropout.
- The user interface hides most complications on group theory and solutions of the kernel space constraint and requires the user only to specify the transformation laws of feature spaces. For instance, a C_8-equivariant convolution operation, mapping a RGB image, identified as three scalar fields, to ten regular feature fields, would be instantiated by:

```
r2_act = Rot2dOnR2(N=8)
feat_type_in = FieldType(r2_act, 3*[r2_act.trivial_repr])
feat_type_out = FieldType(r2_act, 10*[r2_act.regular_repr])
conv_op = R2Conv(feat_type_in, feat_type_out,
kernel_size=5)
```

Implementation details (5)

Everything the user has to do is to specify that

- the group C_8 acts on $\mathbb{R}^2$ by rotating it (line 1)
- to define the types $\rho_{in} = \bigoplus_{i=1}^{3} 1$ and $\rho_{out} = \bigoplus_{i=1}^{10} \rho_{\text{reg}}^{C_4}$ of the input and output feature fields (lines 2 and 3)
- which are subsequently passed to the constructor of the steerable convolution (line 4).

```
r2_act = Rot2dOnR2(N=8)
feat_type_in = FieldType(r2_act, 3*[r2_act.trivial_repr])
feat_type_out = FieldType(r2_act, 10*[r2_act.regular_repr])
conv_op = R2Conv(feat_type_in, feat_type_out,
kernel_size=5)
```

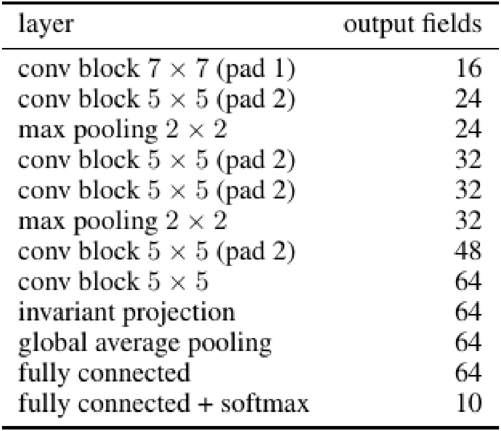
Experiments with various equivariant CNNs can be found in a dedicated repository athttps://github.com/QUVA-Lab/e2cnn_experiments

Part 2. More Details of the Implementation and Performance

Предварительные замечания

- Главное: учет симметрий открывает возможности для огромного числа различных архитектур
  - выбор группы; представлений для слоев; возможных сужений или расширений группы симметрий от слоя к слою
  - выбор числа мод для для эквивариантного ядра (фильтра); выбор радиального профиля:
    - параметрический: гауссов ($R \in \mathbb{Z}$ и σ) (Weiler et al.)
    - произвольно-дискретный (=обучаемые веса; Worral et al.)
  - способ дискретизации (сэмплинг, Gaussian blur)
  - выбор нелинейностей
- только в основной работе Weiler et al. проделана огромная работа по сравнению **57** различных моделей
- + обычный выбор общей архитектуры и гиперпараметров

Общая архитектура экспериментальных сетей



layer	output fields
conv block 7 × 7 (pad 1)	16
conv block 5 × 5 (pad 2)	24
max pooling 2 × 2	24
conv block 5 × 5 (pad 2)	32
conv block 5 × 5 (pad 2)	32
max pooling 2 × 2	32
conv block 5 × 5 (pad 2)	48
conv block 5 × 5	64
invariant projection	64
global average pooling	64
fully connected	64
fully connected + softmax	10

- Each convolution block includes a convolution layer, batch-normalization and a nonlinearity.
- The width is expressed as the number of fields of a regular C_{16} model

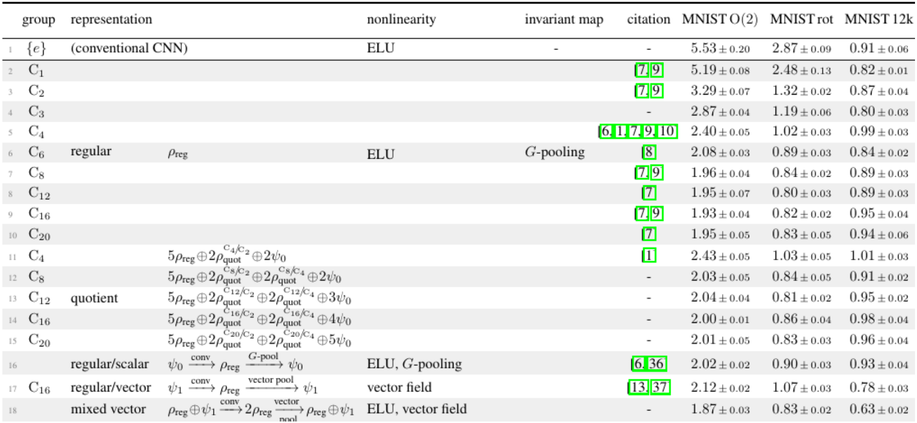
Некоторые детали архитектуры и гиперпараметров

- Training is performed with a batch size of 64 samples, using the Adam optimizer.
- The learning rate is initialized to 10^{-3} and decayed exponentially by a factor of 0.8 per epoch, starting after a burn in phase of 10 epochs.
- In all experiments: steerable bases with Gaussian radial profiles of width $\sigma = 0.6$ for all except the outermost ring where we use $\sigma = 0.4$.
    - The strong cutoff in the rings of maximal radius is motivated by empirical observation that these rings introduce a relatively high equivariance error for higher frequencies.
- a strong bandlimiting policy was applied which permits frequencies up to $0, 2, 2$ for radii $0, 1, 2$ in a 5×5 kernel and up to $0, 2, 3, 2$ for radii $0, 1, 2, 3$ in a 7×7 kernel.
- In order to not disadvantage models with lower levels of equivariance and since it would be done in real scenarios we train all models using augmentation by the transformations present in the corresponding dataset.

Некоторые детали экспериментов

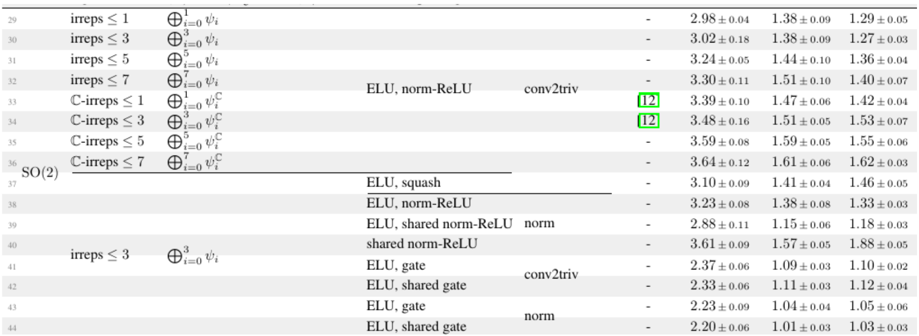
- Table 3 Weiler et al. shows the test errors of **57** different models on the three MNIST variants (MNIST O(2); MNIST rot; MNIST 12k)
- The statistics of each entry are averaged over (at least) 6 samples.
- In order to not disadvantage models with lower levels of equivariance and since it would be done in real scenarios we train all models using augmentation by the transformations present in the corresponding dataset.
- All models apply some form of ***invariant mapping*** to scalar fields followed by spatial pooling after the last convolutional layer such that the predictions are guaranteed to be invariant under the equivariance group of the model.

Some parts of the main Table with the results



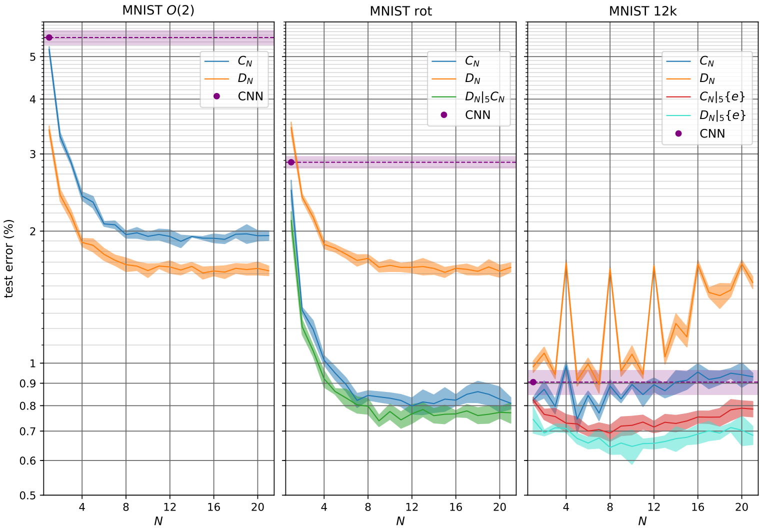
	group	representation		nonlinearity	invariant map	citation	MNIST O(2)	MNIST rot	MNIST 12k
1	$\{e\}$	(conventional CNN)		ELU	-	-	5.53 ± 0.20	2.87 ± 0.09	0.91 ± 0.06
2	C_1	regular	ρ_{reg}	ELU	G-pooling	[7, 9]	5.19 ± 0.08	2.48 ± 0.13	0.82 ± 0.01
3	C_2					[7, 9]	3.29 ± 0.07	1.32 ± 0.02	0.87 ± 0.04
4	C_3					-	2.87 ± 0.04	1.19 ± 0.06	0.80 ± 0.03
5	C_4					[6, 1, 7, 9, 10]	2.40 ± 0.05	1.02 ± 0.03	0.99 ± 0.03
6	C_6					[8]	2.08 ± 0.03	0.89 ± 0.03	0.84 ± 0.02
7	C_8					[7, 9]	1.96 ± 0.04	0.84 ± 0.02	0.89 ± 0.03
8	C_{12}					[7]	1.95 ± 0.07	0.80 ± 0.03	0.89 ± 0.03
9	C_{16}					[7, 9]	1.93 ± 0.04	0.82 ± 0.02	0.95 ± 0.04
10	C_{20}					[7]	1.95 ± 0.05	0.83 ± 0.05	0.94 ± 0.06
11	C_4	quotient	$5\rho_{\text{reg}} \oplus 2\rho_{\text{quot}}^{C_4/C_2} \oplus 2\psi_0$			[1]	2.43 ± 0.05	1.03 ± 0.05	1.01 ± 0.03
12	C_8		$5\rho_{\text{reg}} \oplus 2\rho_{\text{quot}}^{C_8/C_2} \oplus 2\rho_{\text{quot}}^{C_8/C_4} \oplus 2\psi_0$			-	2.03 ± 0.05	0.84 ± 0.05	0.91 ± 0.02
13	C_{12}		$5\rho_{\text{reg}} \oplus 2\rho_{\text{quot}}^{C_{12}/C_2} \oplus 2\rho_{\text{quot}}^{C_{12}/C_4} \oplus 3\psi_0$			-	2.04 ± 0.04	0.81 ± 0.02	0.95 ± 0.02
14	C_{16}		$5\rho_{\text{reg}} \oplus 2\rho_{\text{quot}}^{C_{16}/C_2} \oplus 2\rho_{\text{quot}}^{C_{16}/C_4} \oplus 4\psi_0$			-	2.00 ± 0.01	0.86 ± 0.04	0.98 ± 0.04
15	C_{20}		$5\rho_{\text{reg}} \oplus 2\rho_{\text{quot}}^{C_{20}/C_2} \oplus 2\rho_{\text{quot}}^{C_{20}/C_4} \oplus 5\psi_0$			-	2.01 ± 0.05	0.83 ± 0.03	0.96 ± 0.04
16	C_{16}	regular/scalar	$\psi_0 \xrightarrow{\text{conv}} \rho_{\text{reg}} \xrightarrow{G\text{-pool}} \psi_0$	ELU, G-pooling		[6, 36]	2.02 ± 0.02	0.90 ± 0.03	0.93 ± 0.04
17		regular/vector	$\psi_1 \xrightarrow{\text{conv}} \rho_{\text{reg}} \xrightarrow{\text{vector pool}} \psi_1$	vector field		[13, 37]	2.12 ± 0.02	1.07 ± 0.03	0.78 ± 0.03
18		mixed vector	$\rho_{\text{reg}} \oplus \psi_1 \xrightarrow{\text{conv}} 2\rho_{\text{reg}} \xrightarrow[\text{pool}]{\text{vector}} \rho_{\text{reg}} \oplus \psi_1$	ELU, vector field		-	1.87 ± 0.03	0.83 ± 0.02	0.63 ± 0.02

Some parts of the main Table with the results (2)



29	SO(2)	irreps ≤ 1	$\bigoplus_{i=0}^{1} \psi_i$	ELU, norm-ReLU	conv2triv	-	2.98 ± 0.04	1.38 ± 0.09	1.29 ± 0.05
30		irreps ≤ 3	$\bigoplus_{i=0}^{3} \psi_i$			-	3.02 ± 0.18	1.38 ± 0.09	1.27 ± 0.03
31		irreps ≤ 5	$\bigoplus_{i=0}^{5} \psi_i$			-	3.24 ± 0.05	1.44 ± 0.10	1.36 ± 0.04
32		irreps ≤ 7	$\bigoplus_{i=0}^{7} \psi_i$			-	3.30 ± 0.11	1.51 ± 0.10	1.40 ± 0.07
33		$\mathbb{C}$-irreps ≤ 1	$\bigoplus_{i=0}^{1} \psi_i^{\mathbb{C}}$			[12]	3.39 ± 0.10	1.47 ± 0.06	1.42 ± 0.04
34		$\mathbb{C}$-irreps ≤ 3	$\bigoplus_{i=0}^{3} \psi_i^{\mathbb{C}}$			[12]	3.48 ± 0.16	1.51 ± 0.05	1.53 ± 0.07
35		$\mathbb{C}$-irreps ≤ 5	$\bigoplus_{i=0}^{5} \psi_i^{\mathbb{C}}$			-	3.59 ± 0.08	1.59 ± 0.05	1.55 ± 0.06
36		$\mathbb{C}$-irreps ≤ 7	$\bigoplus_{i=0}^{7} \psi_i^{\mathbb{C}}$			-	3.64 ± 0.12	1.61 ± 0.06	1.62 ± 0.03
37		irreps ≤ 3	$\bigoplus_{i=0}^{3} \psi_i$	ELU, squash		-	3.10 ± 0.09	1.41 ± 0.04	1.46 ± 0.05
38				ELU, norm-ReLU	norm	-	3.23 ± 0.08	1.38 ± 0.08	1.33 ± 0.03
39				ELU, shared norm-ReLU		-	2.88 ± 0.11	1.15 ± 0.06	1.18 ± 0.03
40				shared norm-ReLU		-	3.61 ± 0.09	1.57 ± 0.05	1.88 ± 0.05
41				ELU, gate	conv2triv	-	2.37 ± 0.06	1.09 ± 0.03	1.10 ± 0.02
42				ELU, shared gate		-	2.33 ± 0.06	1.11 ± 0.03	1.12 ± 0.04
43				ELU, gate	norm	-	2.23 ± 0.09	1.04 ± 0.04	1.05 ± 0.06
44				ELU, shared gate		-	2.20 ± 0.06	1.01 ± 0.03	1.03 ± 0.03

Test errors of C_N and D_N regular steerable CNNs for different orders N for all three MNIST variants.



Некоторые выводы

- **All equivariant models outperform the non-equivariant CNN baseline.**
- Overall, regular steerable CNNs with C_N, D_N perform very well **(better than $SO(2)$, $O(2)$)**. Для меня это неожиданность!
  - The reason for this is that feature vectors, transforming under regular representations, can encode any function on the group (??).
- оптимальная дискретная группа не связана с изометрией решетки (т.е. скорее всего может быть выше, чем C_6 для гексогональной решетки)