Clifford Algebra Representations for Deep Learning

CaLISTA Workshop

David Ruhe - September 4, 2024

About Me

- PhD-student at AMLab (University of Amsterdam)
  - AI4Science
  - Generative Models
  - Time-Series
  - Geometric Deep Learning

AMLAB
Amsterdam
Machine Learning Lab

Ph.D. Student at the University of Amsterdam

Overview

- Clifford Algebra
- Clifford Group Equivariant Neural Networks
- Clifford Group Equivariant Simplicial Message Passing
- Clifford-Steerable CNNs

Clifford Group Equivariant Neural Networks

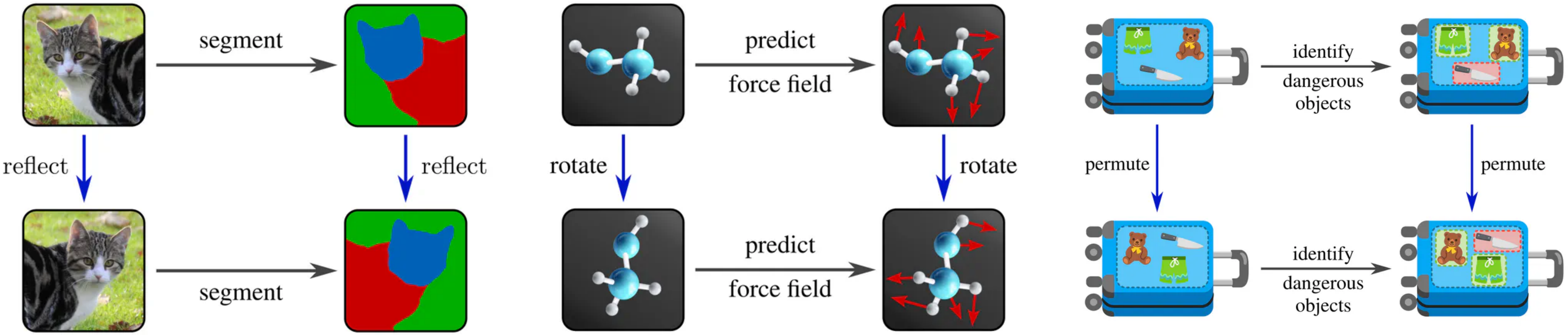
David Ruhe
University of Amsterdam
AI4Science Lab, AMLab, API
david.ruhe@gmail.com

Johannes Brandstetter
Microsoft Research
AI4Science
johannesb@microsoft.com

Patrick Forré
University of Amsterdam
AI4Science Lab, AMLab
p.d.forre@uva.nl

Introduction

Equivariant Neural Networks



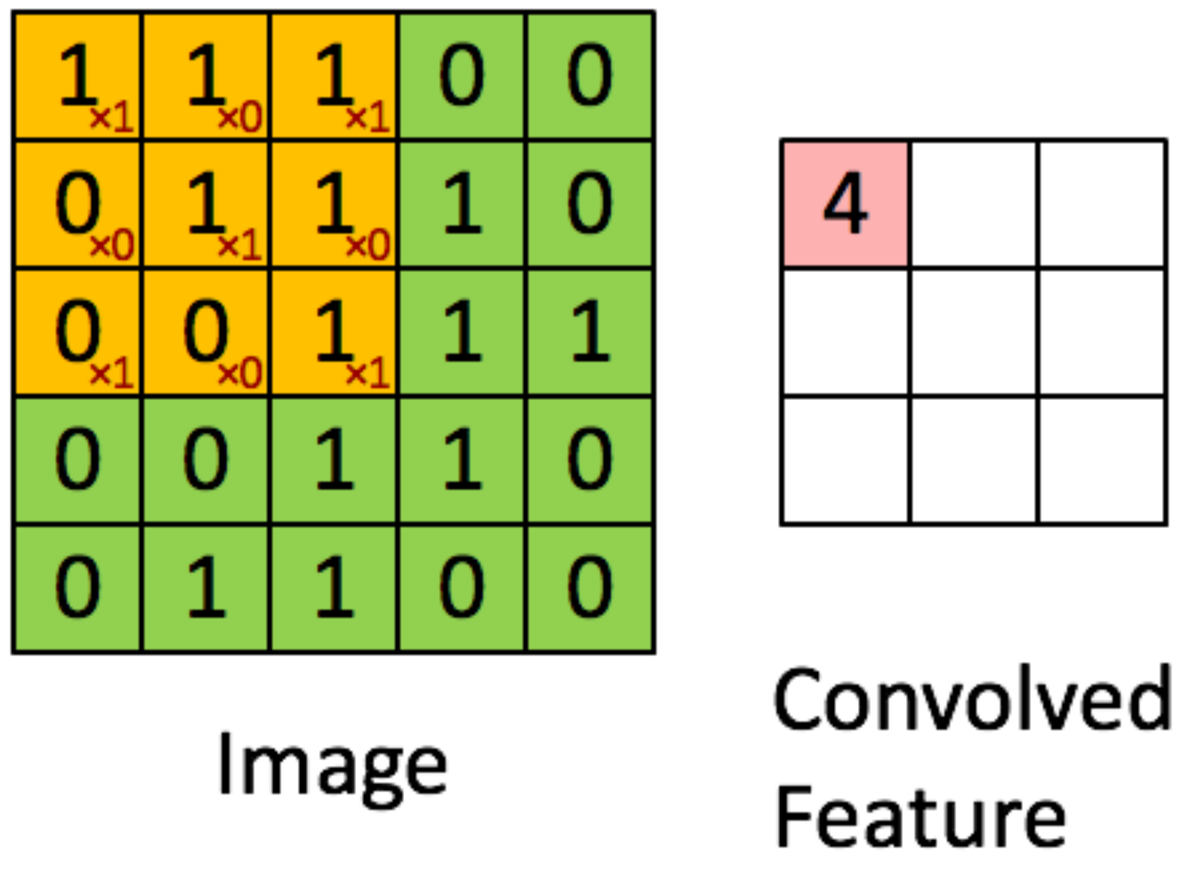
- Group equivariance stimulates *robust* and *reliable* results.
- $w \in G : \rho(w)\phi = \phi\rho(w)$

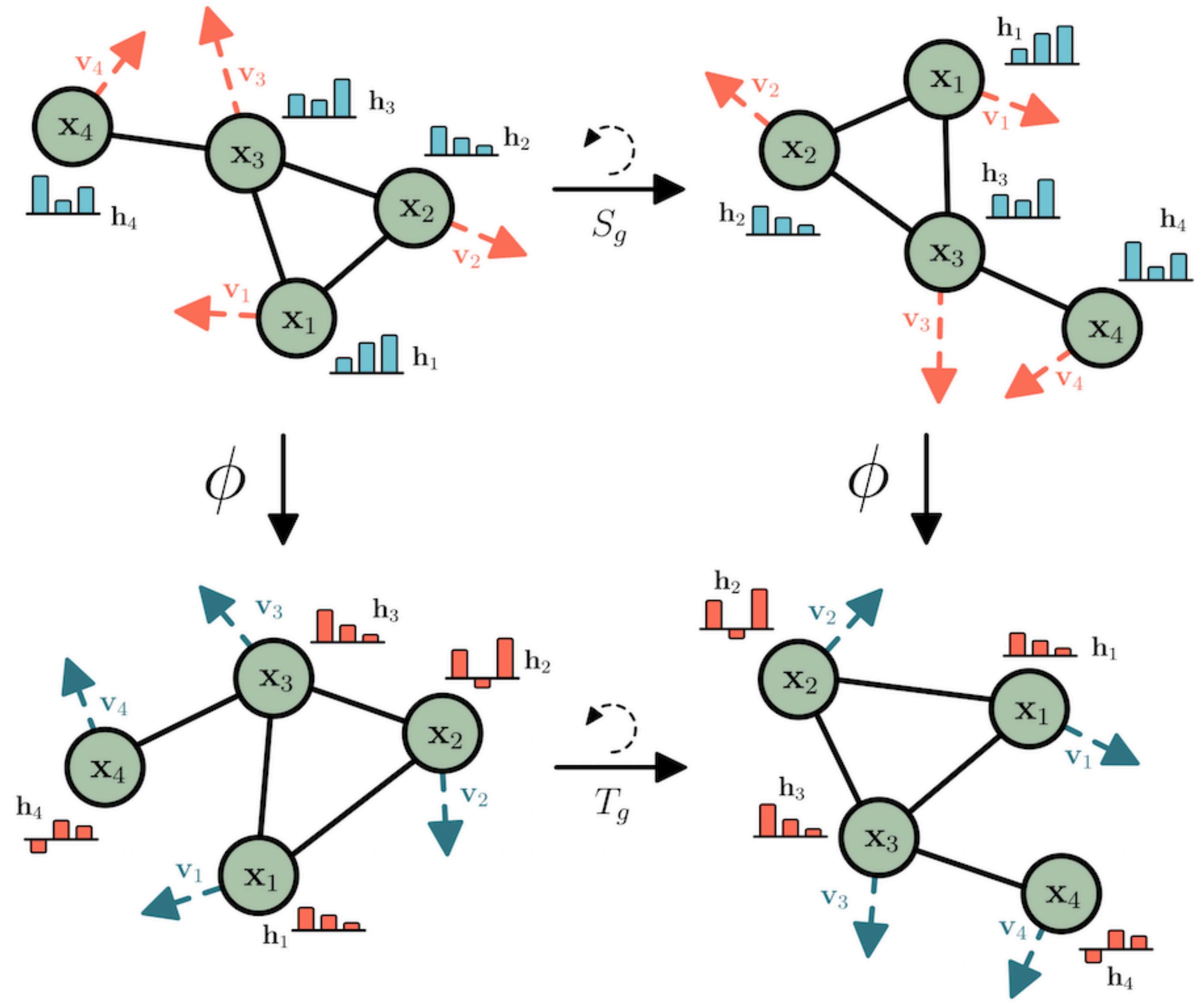
- Images by Maurice Weiler

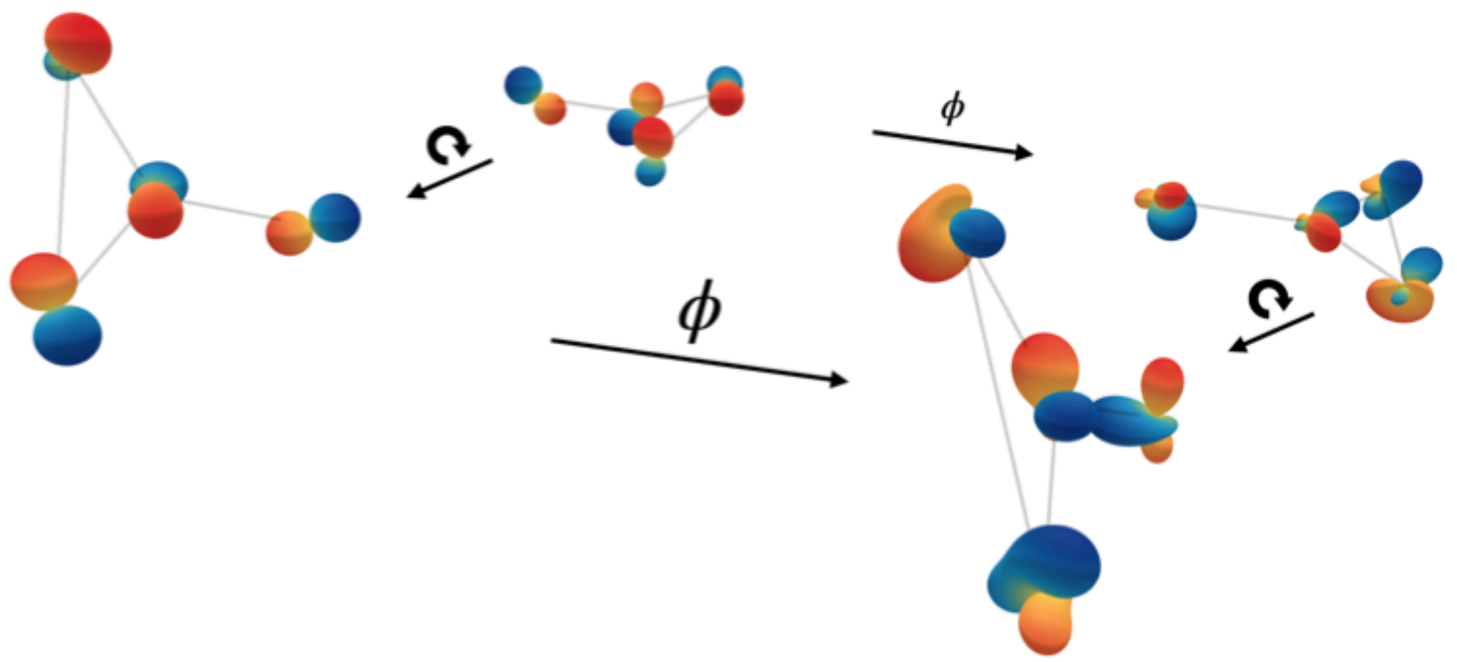
Introduction

Equivariant (Graph) Networks: Categorization

- Group convolutions (LieConv, B-spline CNNs).
  - Integral over a group - computationally intensive.
- Scalarization methods (EGNN, GVP, VN).
  - Operate almost exclusively with invariant (scalar) features.
  - Restricted expressivity.
- *E(3)-NN* based methods (TFN, SEGNN).
  - Tensor products of Wigner-D representations decomposed into irreps using Clebsch-Gordan coefficients.
  - Operate on spherical harmonics basis.
  - Not trivially extended to other dimensions or groups than O(3).

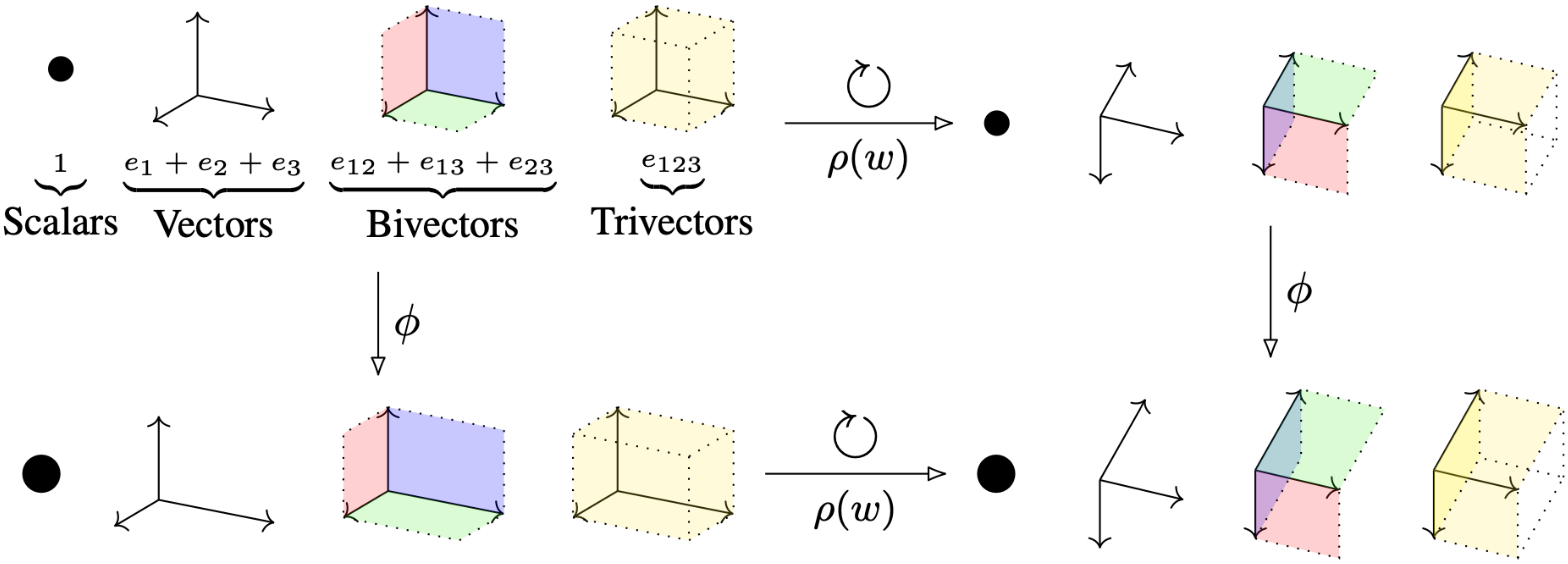






Introduction

Clifford Group Equivariant Networks



The Clifford Algebra

Introduction

- Also known as *geometric algebra*.
- Algebraic representation and manipulation of geometric concepts.
- Generalization of the exterior algebra.
- Inclusion of complex numbers and quaternions.
- Coordinate / Dimension independent.
- Applications in robotics, computer graphics, signal processing, physics, biology, etc.

The Clifford Algebra

Why Deep Learning?

- Some indications CA data representations + CA weights yields more efficient learning + generalization properties.
  - Similar to complex neural networks.
- Can represent certain physics quantities through e.g. bivectors.
- Equivariance w.r.t. several groups in several dimensions (O(3), SO(3), O(2), O(1, 3), E(3), etc.
  - Translations (PGA), conformal group.
- Equivariant **multiplicative** operation (geometric product).
  - No need for spherical harmonics, CG coefficients, etc. Space is bounded.

The Clifford Algebra

Bilinear Forms

- Geometry starts with a notion of *distance*. We introduce a *bilinear form*
  - $\langle \cdot , \cdot \rangle : V \times V \to F$
  - Distance between two vectors: $||x - y||^2 = \langle x - y, x - y \rangle$
  - Angles: $\theta_{xy} = \arccos \frac{\langle x, y \rangle}{||x|| \, ||y||}$

The Clifford Algebra

Orthogonal Group

- $\langle \cdot , \cdot \rangle : V \times V \to F$
- The *group* $O(n)$ contains all linear transformations that preserve the bilinear form.
  - $O(n) := \left\{ R \in \mathrm{GL}(n) \mid \forall u, v \in V : \langle u, v \rangle = \langle Ru, Rv \rangle \right\}$
- These generalize to *non-Euclidean* metrics as found in, e.g., special relativity.
  - For example, in the Euclidean case we had $\langle v, w \rangle = v^\top \begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix} w$.
  - In special relativity, we can use $\langle v, w \rangle = v^\top \begin{bmatrix} 1 & & & \\ & -1 & & \\ & & -1 & \\ & & & -1 \end{bmatrix} w$
  - The orthogonal group of such a space is $O(1,3)$, defined analogously to the Euclidean case.

The Clifford Algebra
Introduction

- Algebra: a vector space (e.g., $\mathbb{R}^3$) with a product.
  - $u, v \in V : uv$ is a valid expression.
  - We **specify** a *product* of two vectors that relates to the inner product (geometry) but does not reduce to a scalar.
- We now axiomatically state
  - $v^2 := \|v\|^2 = \langle v, v \rangle$ enforced relation to preserve geometry.
  - $(u+v)^2 = u^2 + v^2 + uv + vu \iff uv + vu = 2\langle u, v \rangle$
  - *We still have to investigate in which space this product lives.*

$v^2 = q(v) \iff (u+v)^2 - u^2 - v^2 = 2b(u,v)$

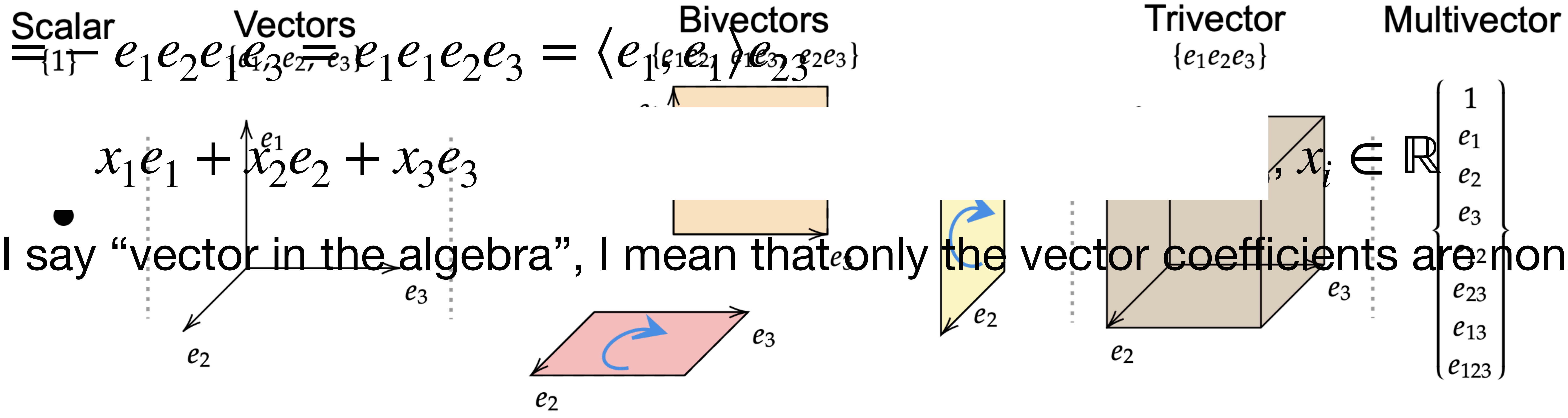
The Clifford Algebra

The Algebra Basis

- Let’s take $\mathbb{R}^3$. Using a basis e_1, e_2, e_3. I.e.,
  - $x \in \mathbb{R}^3 : x = x_1e_1 + x_2e_2 + x_3e_3$
- We can create the $\mathrm{Cl}(\mathbb{R}^3, \langle \cdot , \cdot \rangle)$ basis
- For orthogonal (basis) vectors, $e_ie_j = -e_je_i$.

$$uv + vu = 2\langle u, v \rangle$$

- $e_{123}e_1 = -e_1e_2e_1e_3 = e_1e_1e_2e_3 = \langle e_1, e_1 \rangle e_{23}$
- $x = x_1e_1 + x_2e_2 + x_3e_3$ …, $x_i \in \mathbb{R}$
- When I say “vector in the algebra”, I mean that only the vector coefficients are nonzero.
-

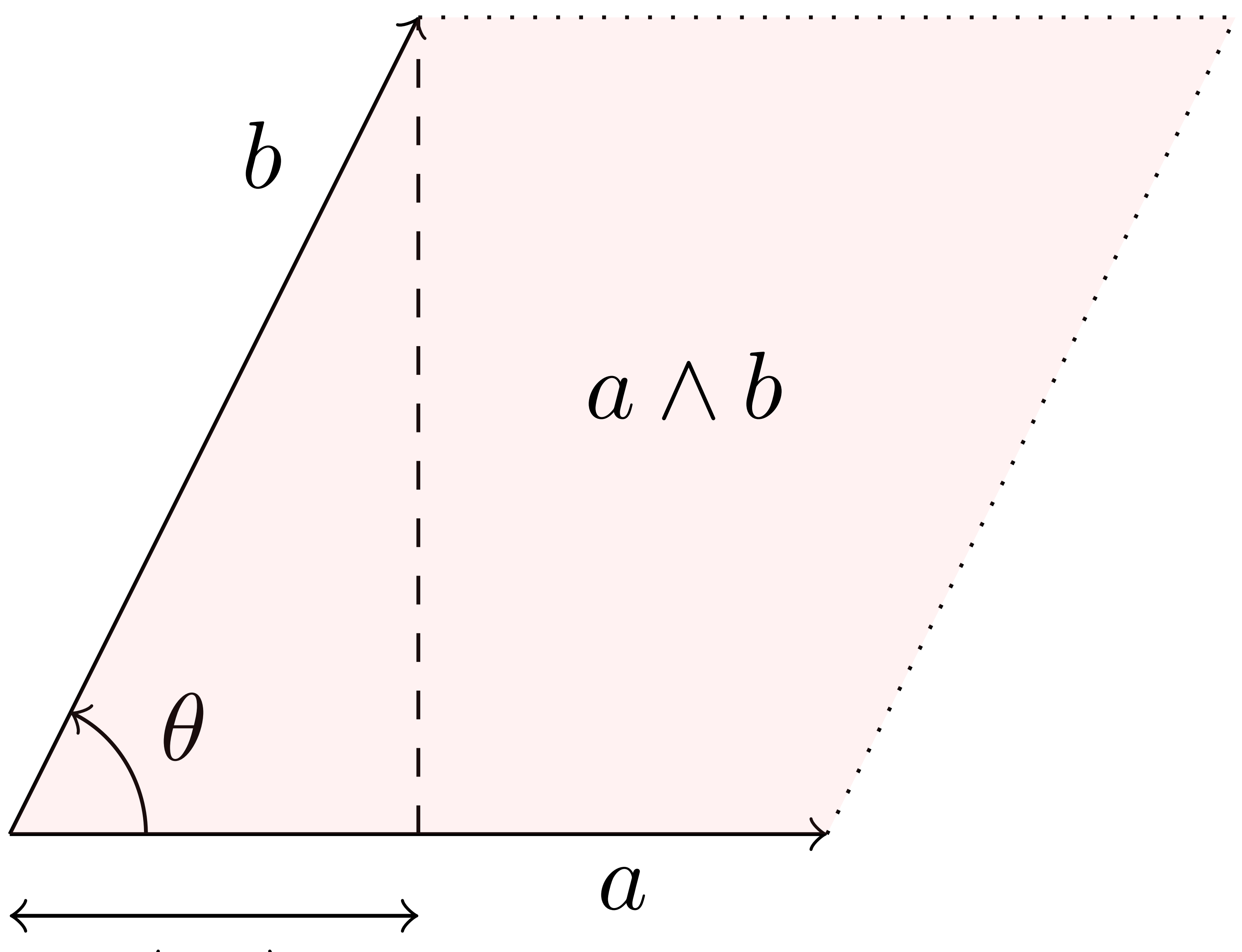


The Clifford Algebra

The Geometric Product

- Let's take two vectors $a, b \in \mathrm{Cl}(\mathbb{R}^2, \langle \cdot , \cdot \rangle)$, $\langle e_i, e_i \rangle = 1$

  $$ab = (a_1e_1 + a_2e_2)(b_1e_1 + b_2e_2) = a_1e_1(b_1e_1 + b_2e_2) + a_2e_2(b_1e_1 + b_2e_2)$$

- 

Theoretical Results

The orthogonal group.

O multivector representation.

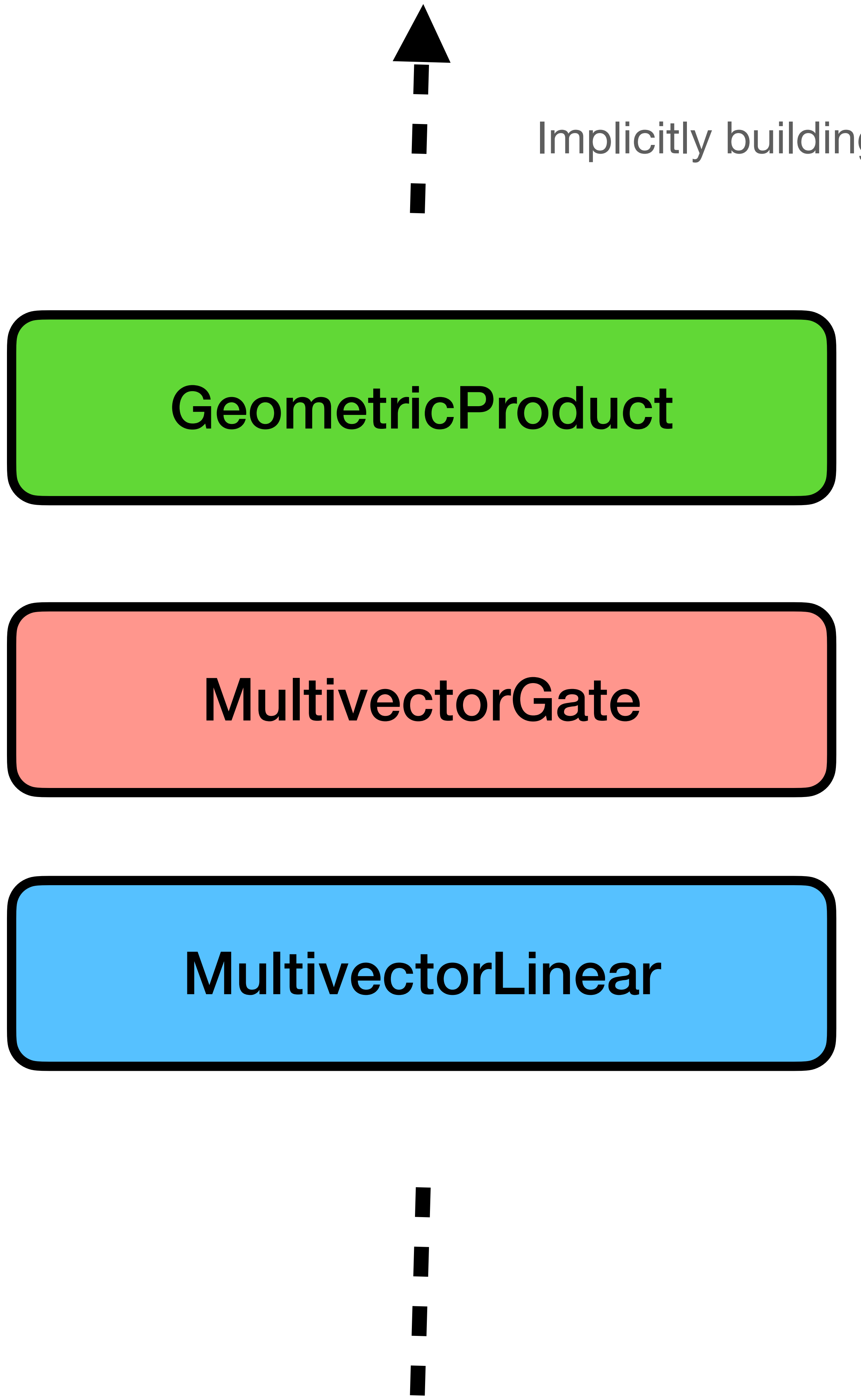
- $\rho(w)$: orthogonal group representation.
- $\rho(w)$ satisfies:

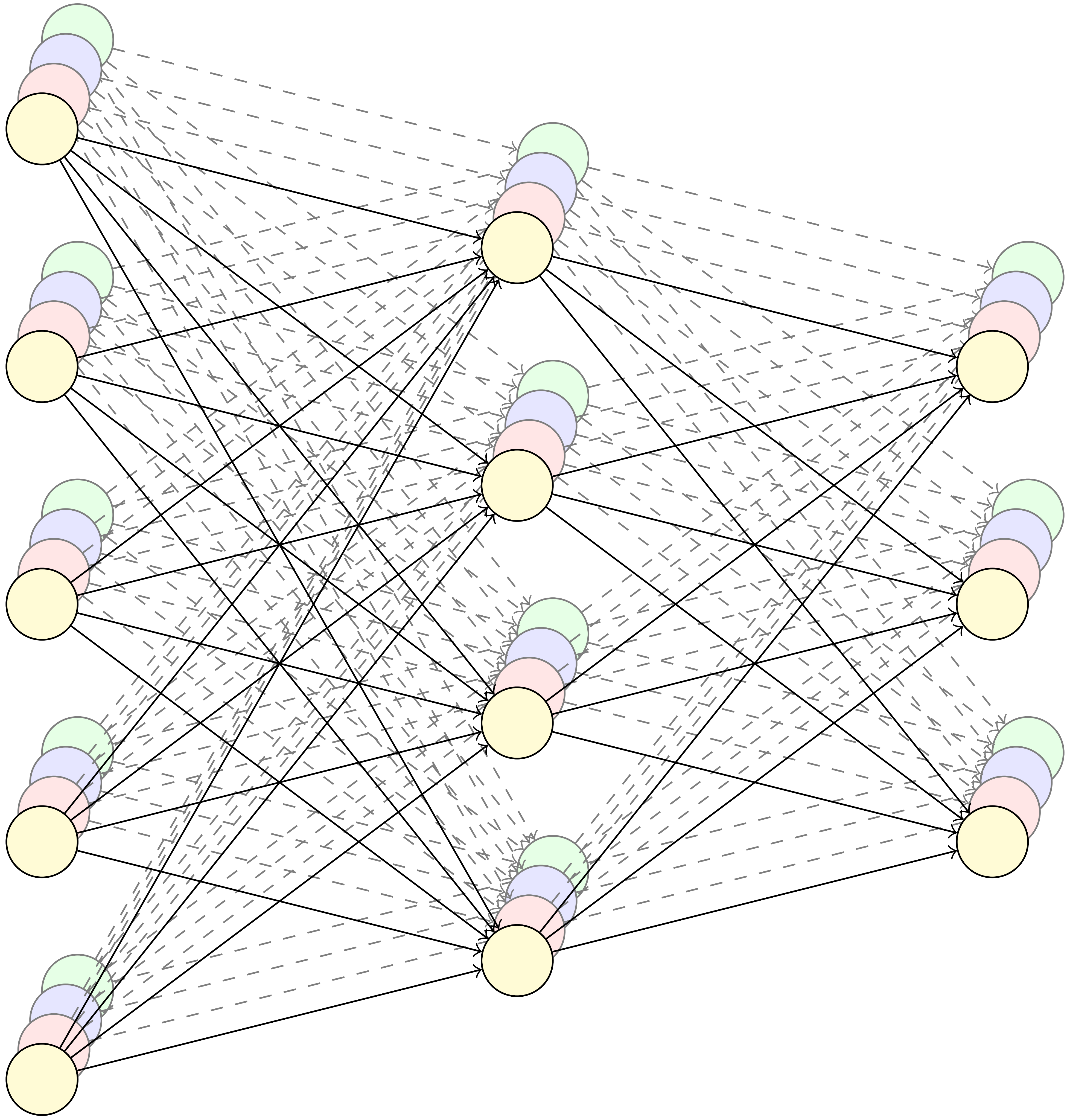
1. $\langle(\rho(w)(x_1), \rho(w)(x_2)\rangle = \langle x_1, x_2\rangle$
2. Additivity: $\rho(w)(x_1 + x_2) = \rho(w)(x_1) + \rho(w)(x_2)$
3. Multiplicativity: $\rho(w)(x_1x_2) = \rho(w)(x_1)\rho(w)(x_2)$
4. Commutes with scalars: $\rho(w)(\alpha \cdot x) = \alpha \cdot \rho(w)(x)$

All geometric product polynomials are O equivariant.

Network Architectures

Implicitly building a big polynomial…

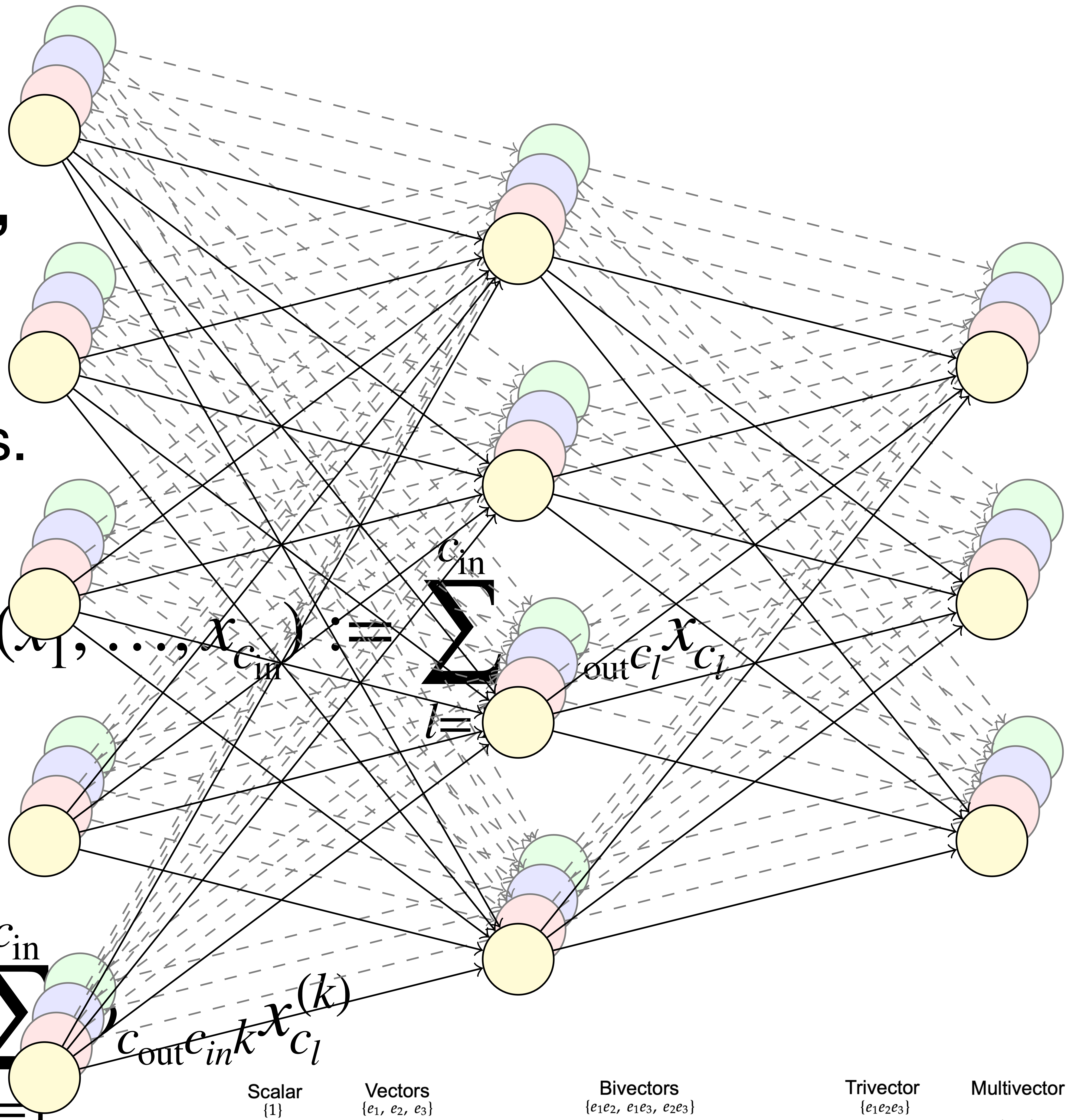


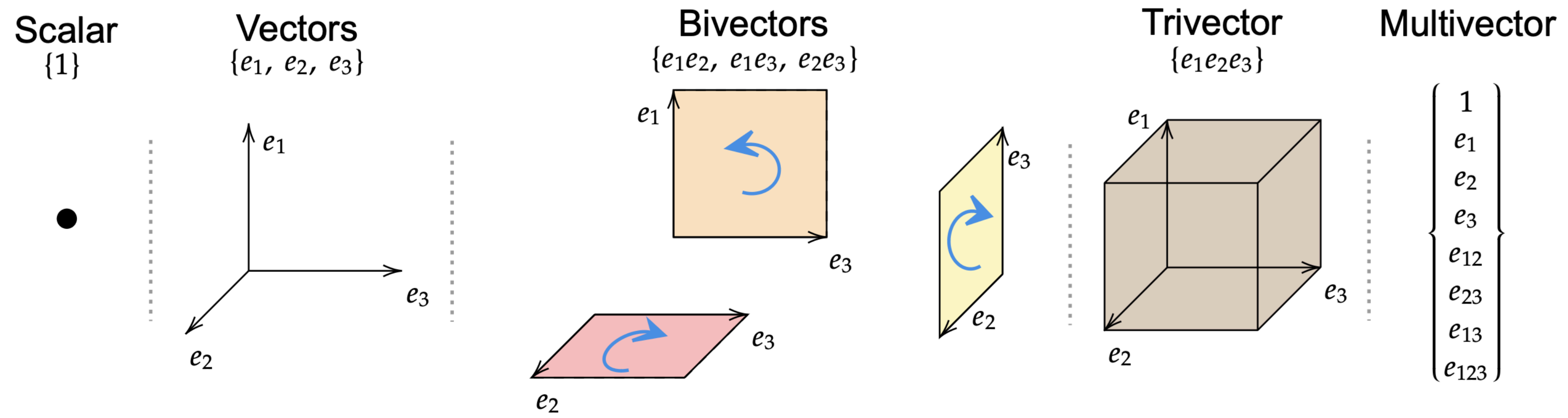


Methodology

Linear Layers “Multivector Neurons”

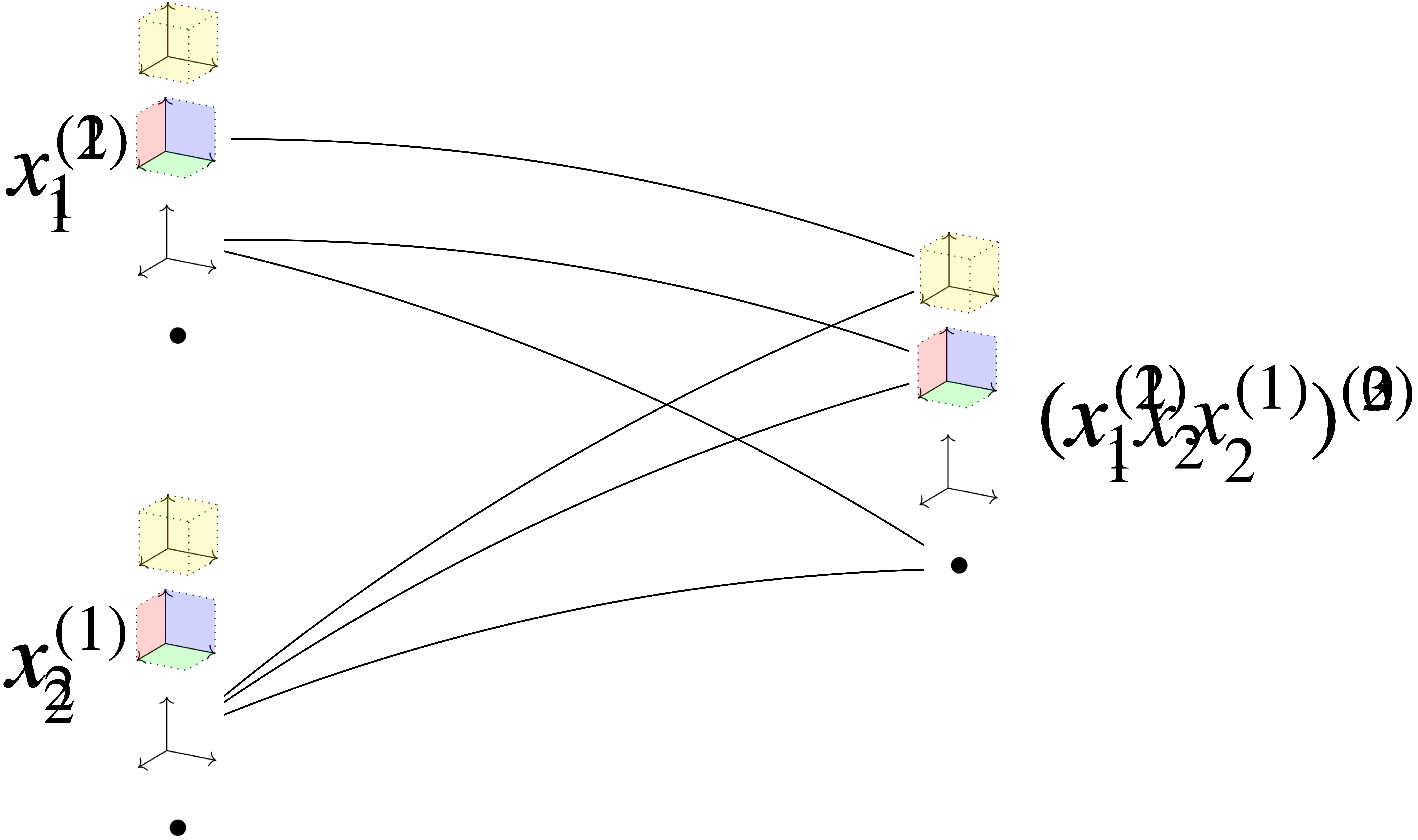
- Let $x_1, \ldots, x_{c_{\text{in}}}$ denote a set of multivectors.
- We can linearly combine them using $T^{\text{lin}}_{\phi_{c_{\text{out}}}}(x_1, \ldots, x_{c_{\text{in}}}) := \sum_{l=1}^{c_{\text{in}}} \phi_{c_{\text{out}} c_l} x_{c_l}$
- $\phi_{c_{\text{out}} c_{\text{in}}} \in \mathbb{R}$
- Or more densely: $T^{\text{lin}}_{\phi_{c_{\text{out}}}}(x_1, \ldots, x_{c_{\text{in}}})^{(k)} := \sum_{l=1}^{c_{\text{in}}} \phi_{c_{\text{out}} c_{in} k} x_{c_l}^{(k)}$



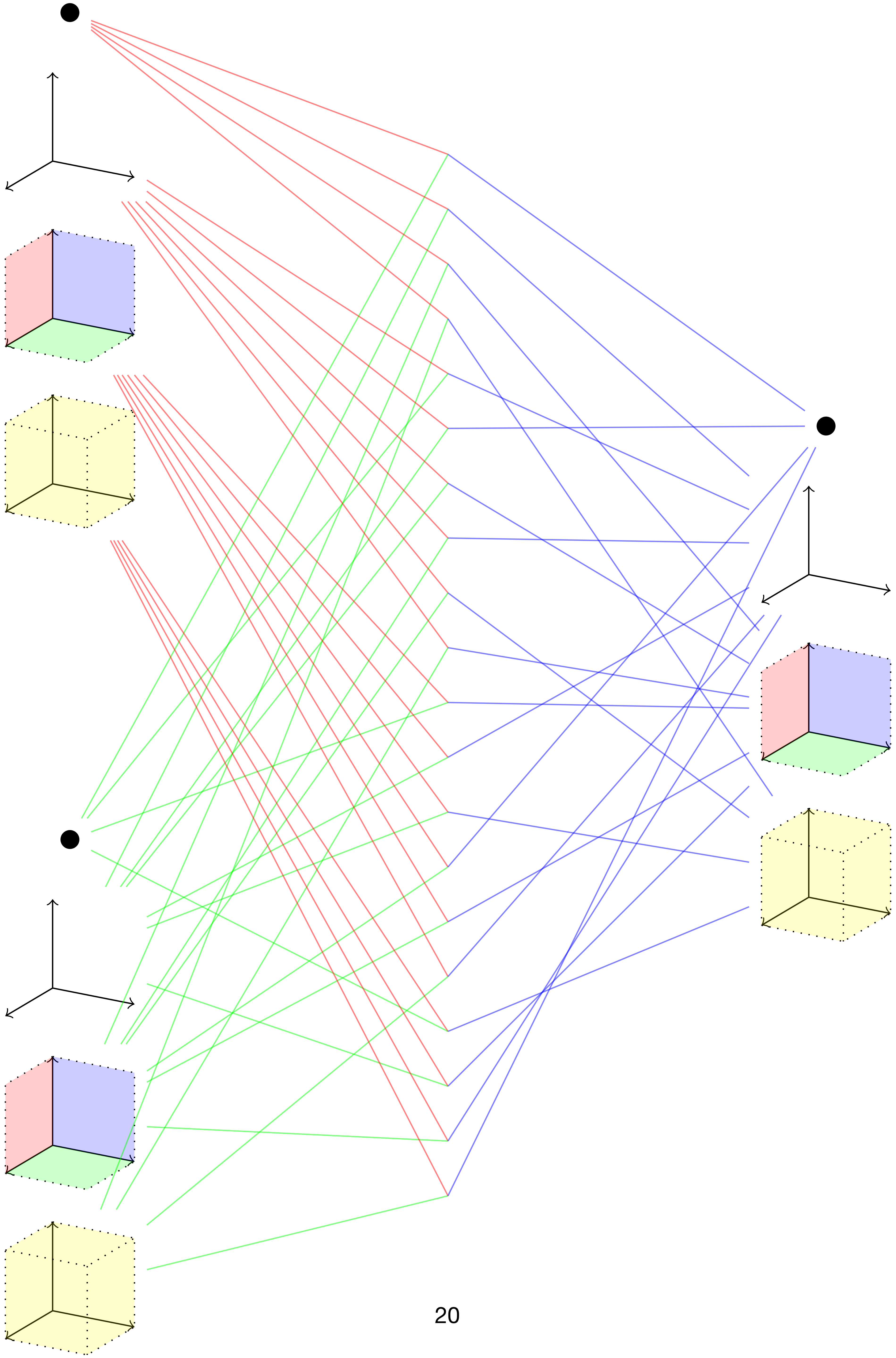


Methodology

Parameterized Geometric Products



$x_2^{(1)}$



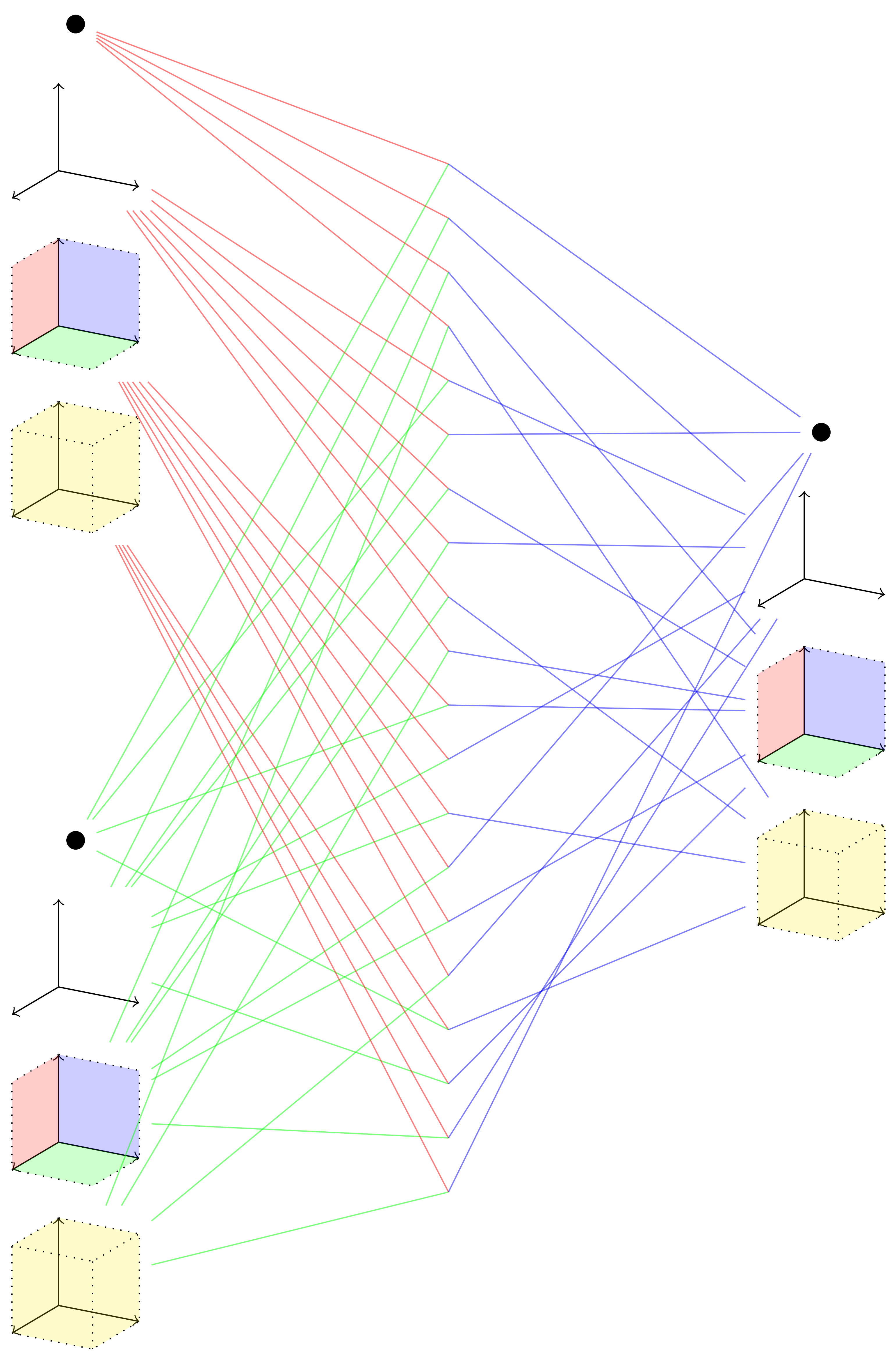
Methodology

Parameterized Geometric Product

- $$P_\phi(x_1, x_2)^{(k)} := \sum_{i=0}^{n} \sum_{j=0}^{n} \phi_{ijk}(x_1^{(i)} x_2^{(j)})^{(k)}$$

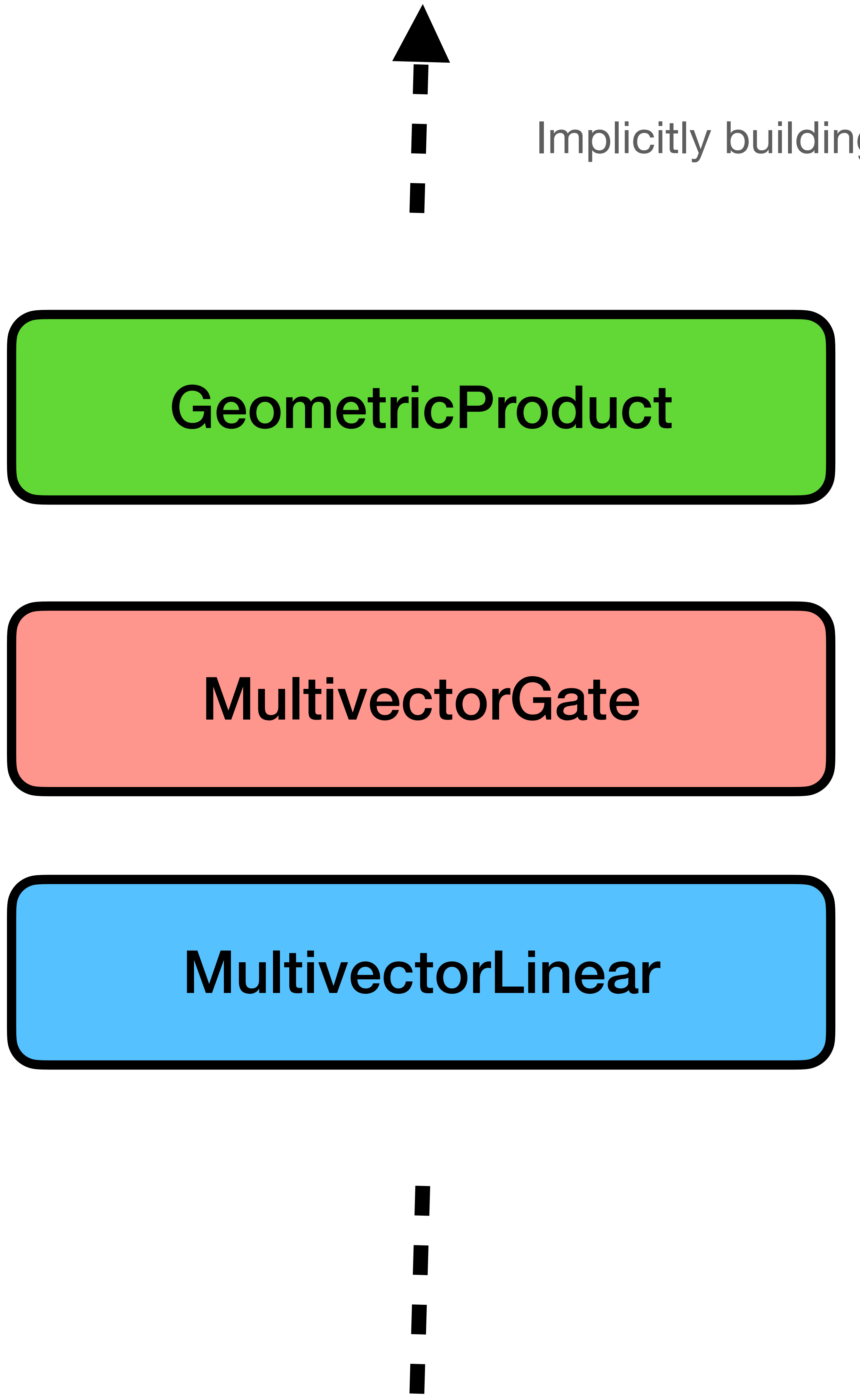
- All products:

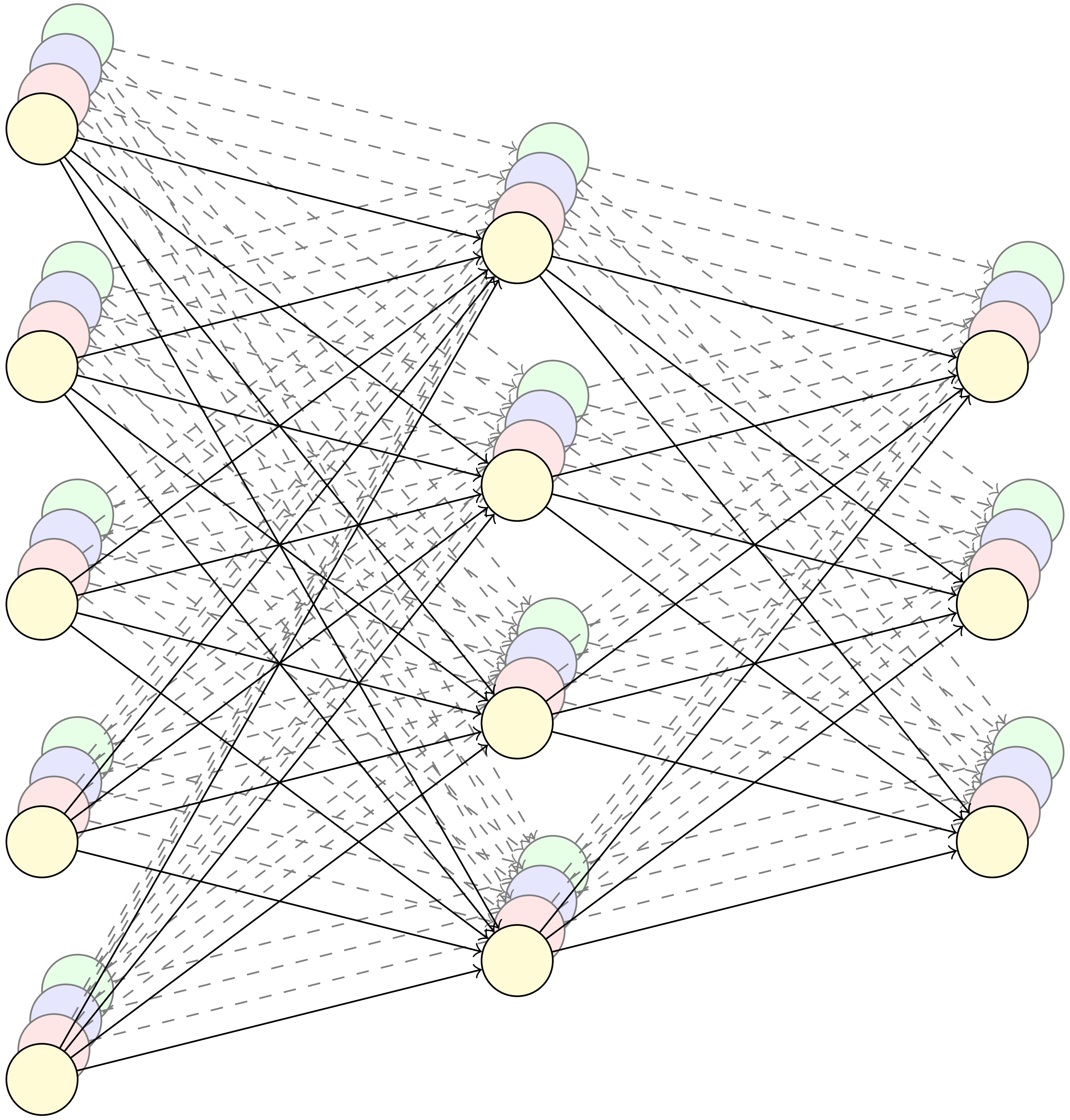
  - $$T^{\text{prod}}(x_1, \ldots, x_{c_{\text{in}}})^{(k)} := \sum_{p=1}^{c_{\text{in}}} \sum_{q=1}^{c_{\text{in}}} P_{\phi_{pq}}(x_p, x_q)^{(k)}$$



Network Architectures

Implicitly building a big polynomial…

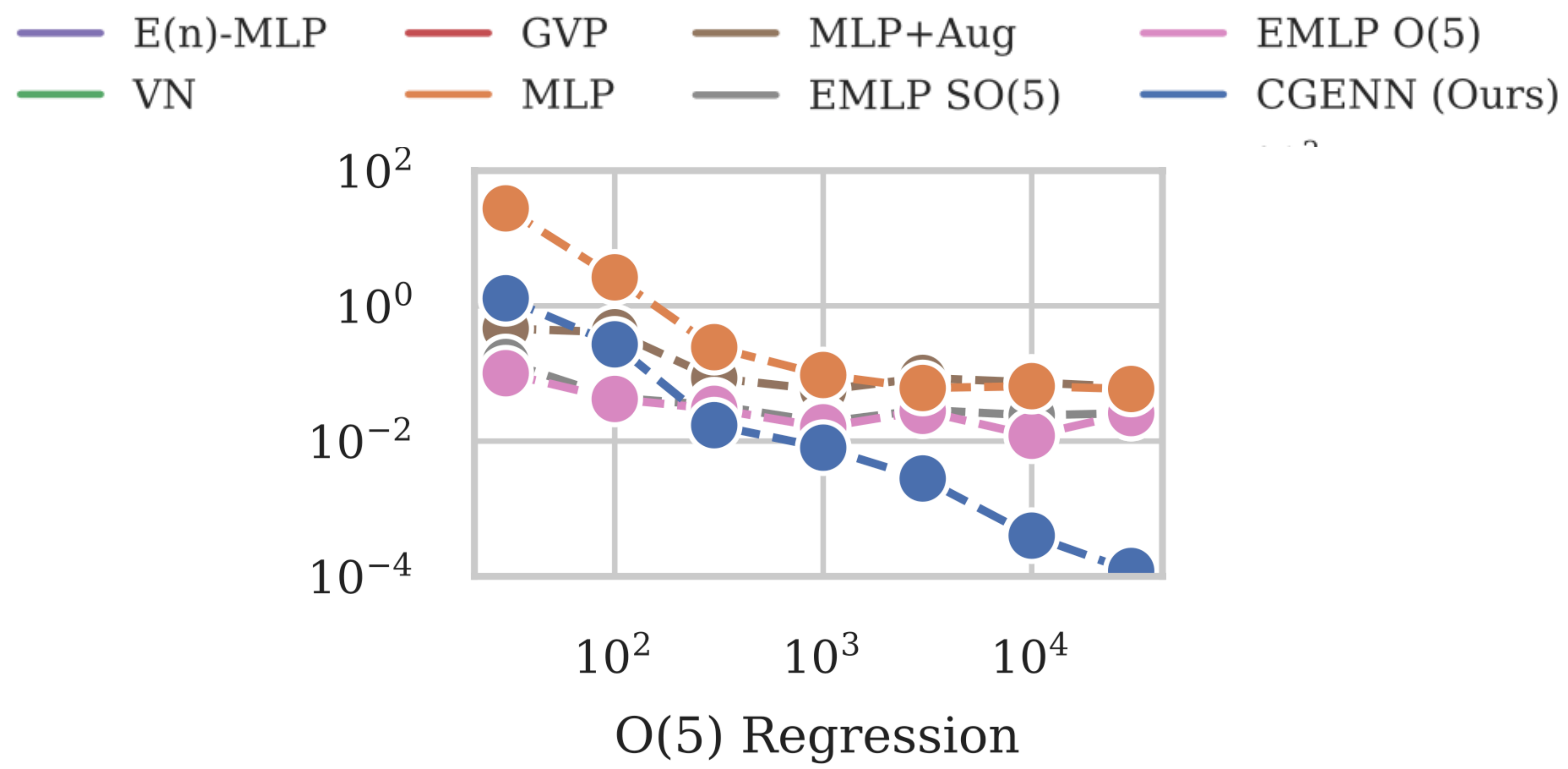




Experiments

O(5) Experiment: Regression

- Taken from Finzi et al., 2022
- Approximate $f(x_1, x_2) := \sin(\|x_1\|) - \|x_2\|^3/2 + \frac{x_1^\top x_2}{\|x_1\|\|x_2\|}$



Experiments

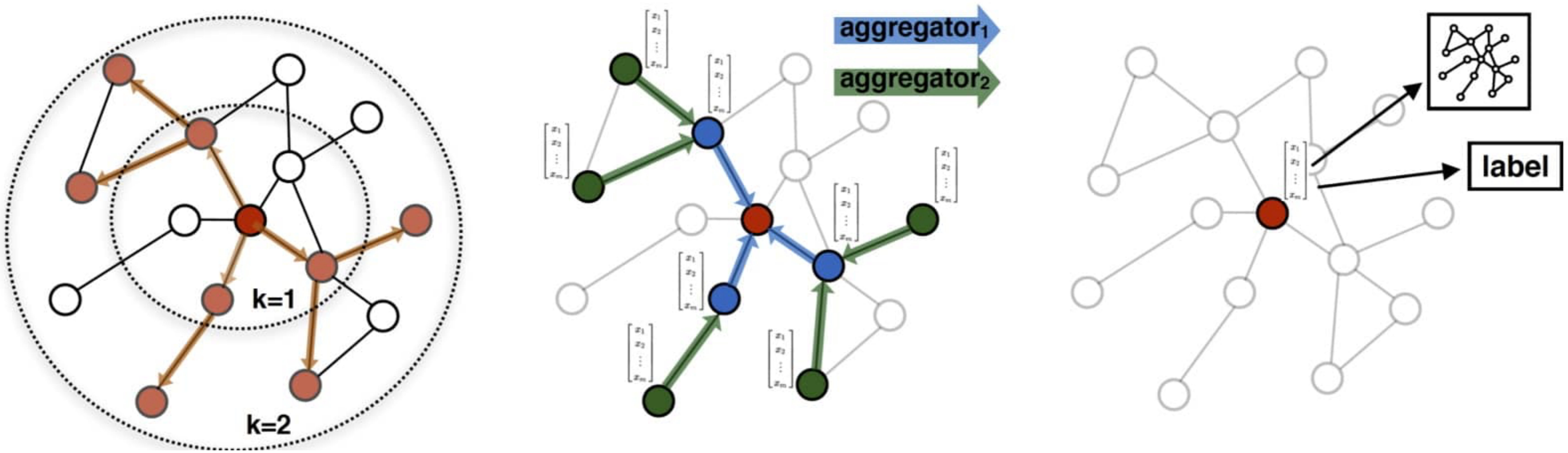
E(3) **Experiment: n-body.**

- A benchmark for simulating physical systems using GNNs.
- Given $n = 5$ charged particles' positions and velocities, estimate their positions after 1000 time-steps.

Experiments

E(3) **Experiment: n-body.**

- A benchmark for simulating physical systems using GNNs.
- Given $n = 5$ charged particles' positions and velocities, estimate their positions after 1000 time-steps.



Algorithm 2 Standard Message Passing

Require: $G = (V, E), \forall v \in V : h^v_{\text{in}}, \phi^m, \phi^h$

$h^v_0 \leftarrow \text{Embed}(h^v_{\text{in}})$
for $\ell = 0, \ldots, L-1$ **do**
  # Message Passing
  $m^v_\ell \leftarrow \text{Agg}_{w \in N(v)} \phi^m(h^v_\ell, h^w_\ell)$
  $h^v_{\ell+1} \leftarrow \phi^h(h^v_\ell, m^v_\ell)$
end for
$h^G \leftarrow \text{Agg}_{v \in V} h^v_L$
$h_{\text{out}} \leftarrow \text{Readout}(h^G)$
return h_{out}

Experiments

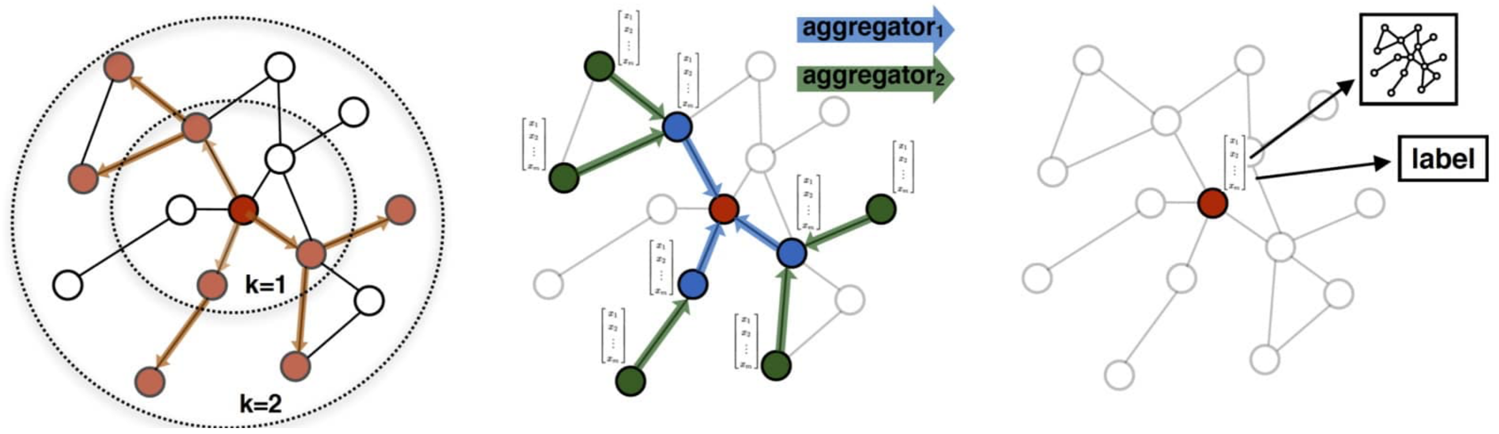
E(3) **Experiment: n-body.**

- A benchmark for simulating physical systems using GNNs.
- Given $n = 5$ charged particles' positions and velocities, estimate their positions after 1000 time-steps.

Method	MSE (↓)
SE(3)-Tr.	0.0244
TFN	0.0155
NMP	0.0107
Radial Field	0.0104
EGNN	0.0070
SEGNN	0.0043
CGENN	$\mathbf{0.0039 \pm 0.0001}$

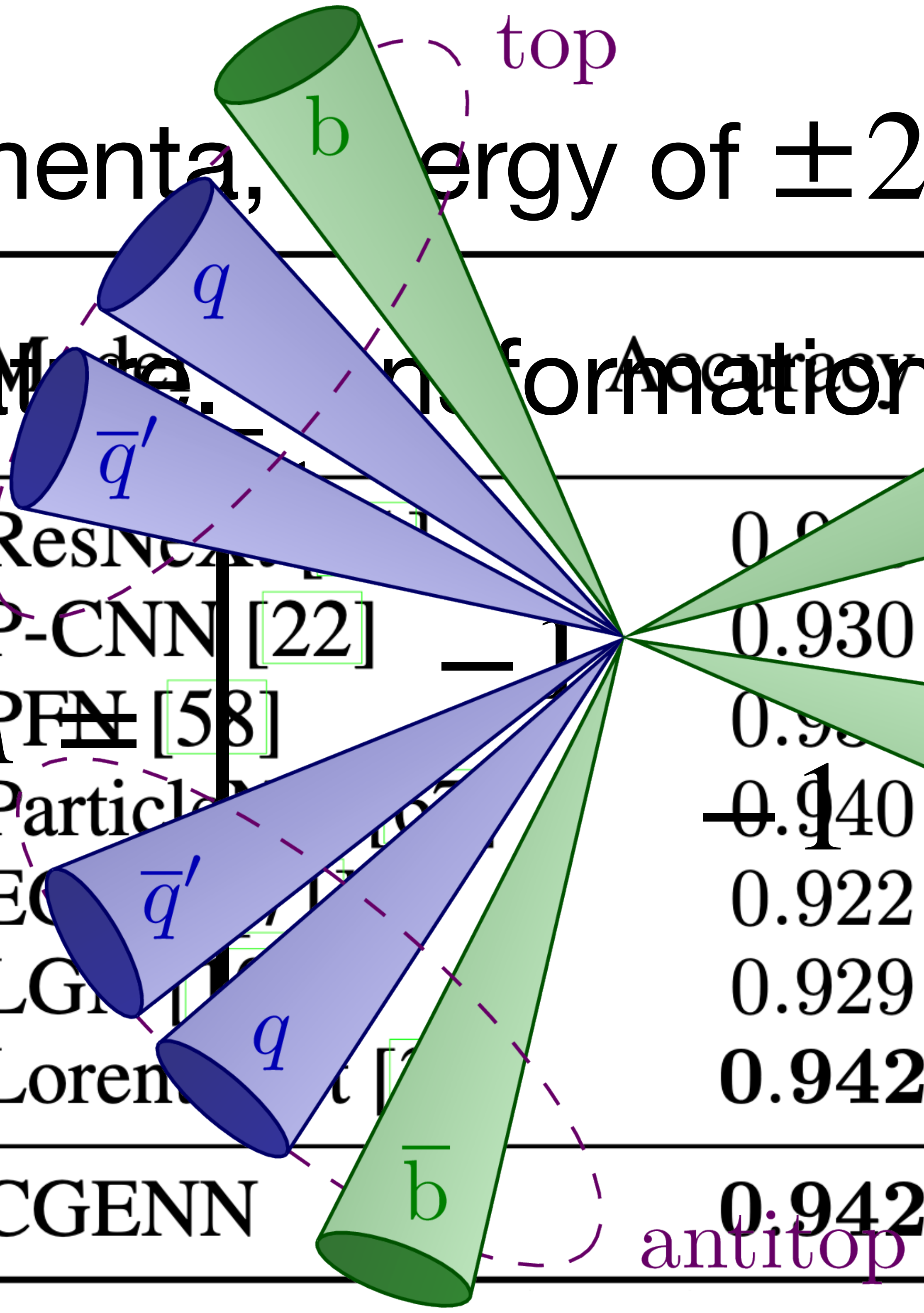
Table 1: Mean-squared error (MSE) on the n-body system experiment.

Experiments



O(1,3) **Experiment: Top Tagging**

- Jet tagging: identifying particle jets generated during collisions.
- Top tagging: identifying whether event produced a top quark.
  - Given: momenta, energy of ±200 particles.
- Relativistic nature: transformations that preserve space-time distances given by O(1,3). $A \in$ ... $\to \{0, 1\}$



Model	Accuracy (↑)	AUC (↑)	$1/\epsilon_B$ (↑) ($\epsilon_S = 0.5$)	$1/\epsilon_B$ (↑) ($\epsilon_S = 0.3$)
ResNeXt [illegible]	[illegible]	0.9837	302	1147
P-CNN [22]	0.930	0.9803	201	759
PFN [58]	[illegible]	0.9819	247	888
Particl[illegible]	0.940	[illegible]	397	1615
E[illegible]	0.922	[illegible]	148	540
LG[illegible]	0.929	0.9640	124	435
Loren[illegible]	**0.942**	**[illegible]**	**498**	**2195**
CGENN	**0.942**	**0.9869**	**500**	**2172**

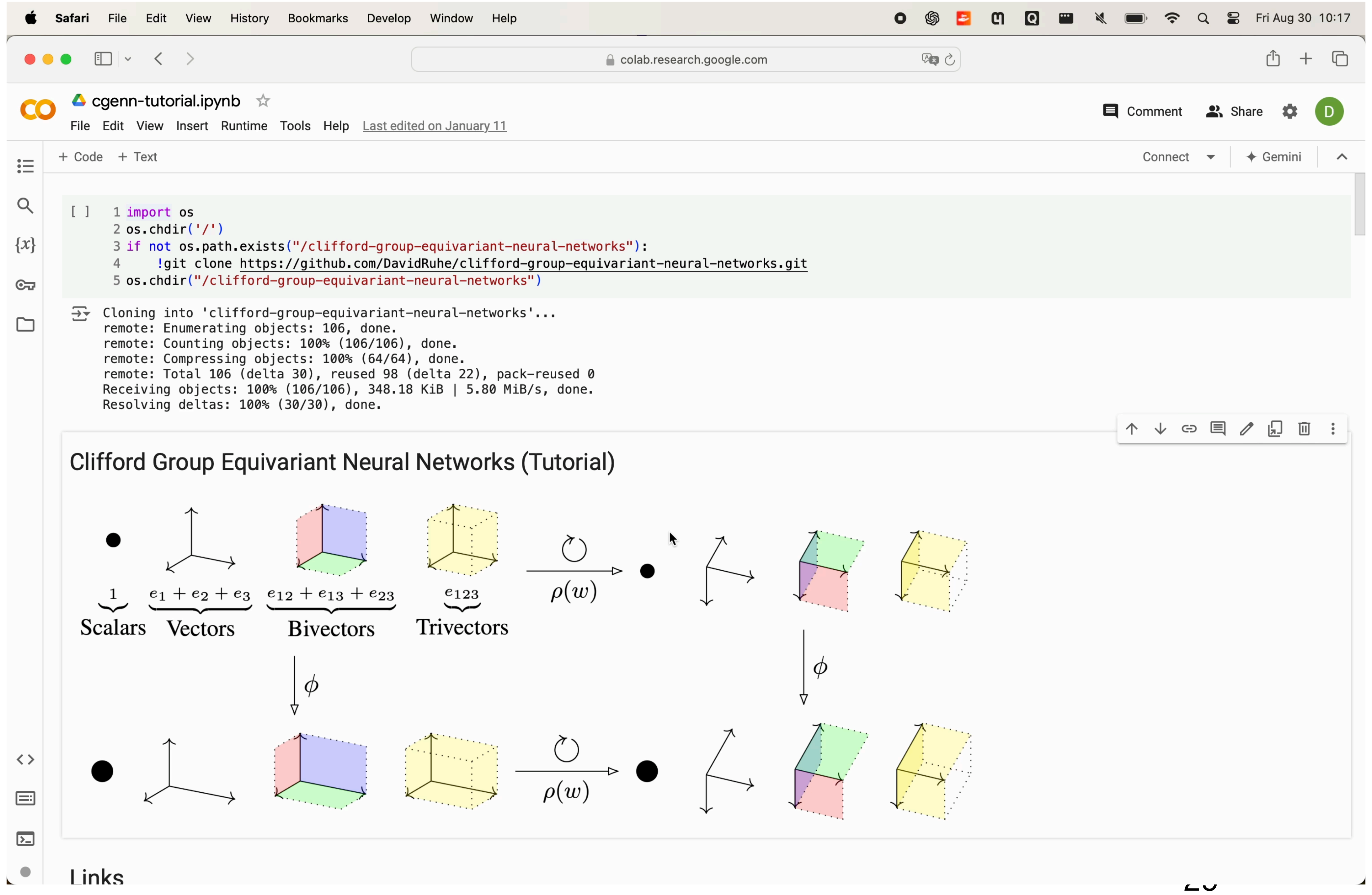
In Conclusion

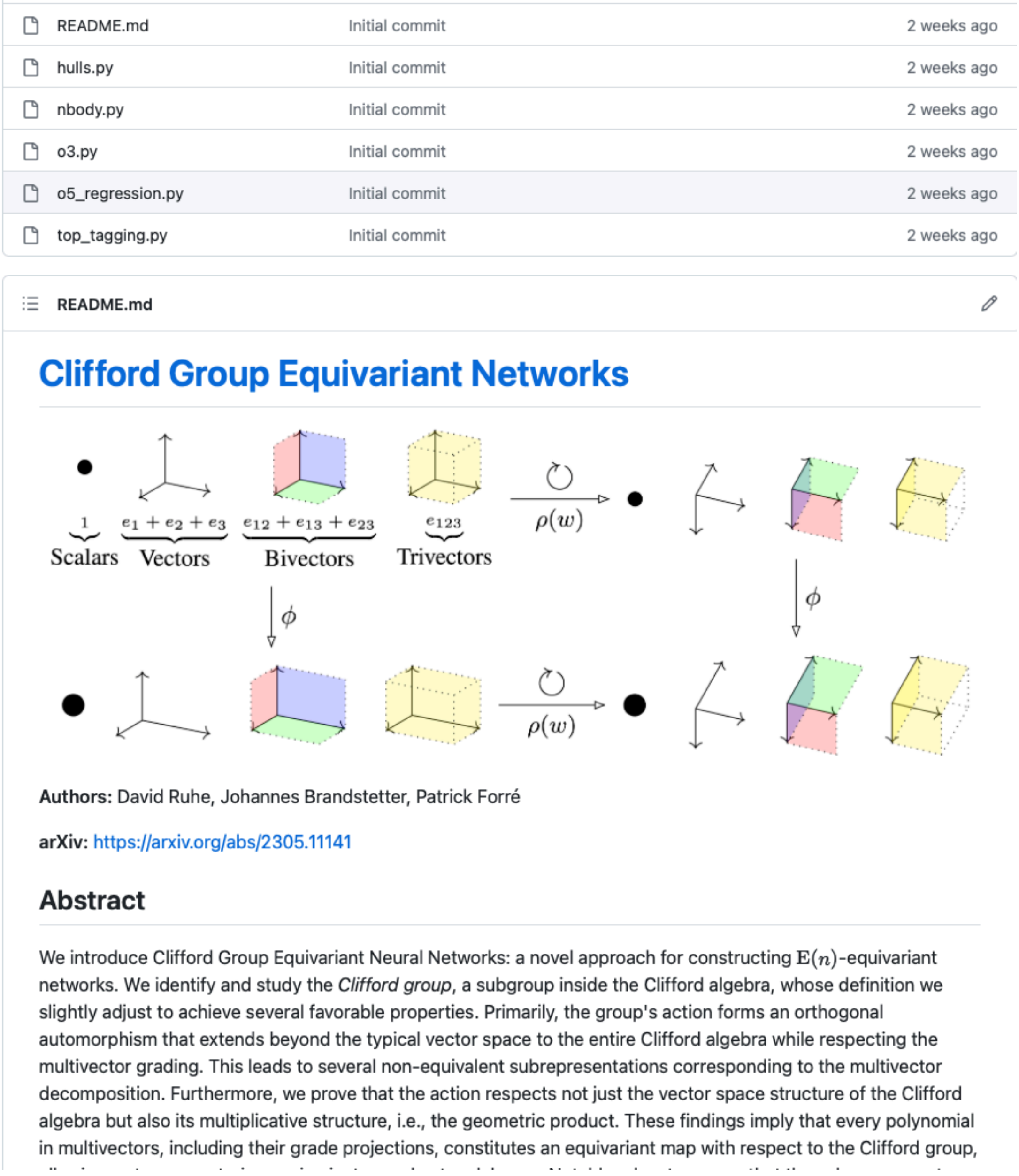
- Equivariant parameterization of neural networks based on *Clifford algebras.*
- Remarkably versatile models: different dimensions and applications.
- Despite that, we match or outperform models specifically designed for certain tasks.

- No need for group convolutions.
- We can directly use higher-order (vector) features instead of scalarized ones.
- CGENNs generalize to quadratic spaces of any dimension.
- No spherical harmonics, CG coefficients, etc.

Final Remarks

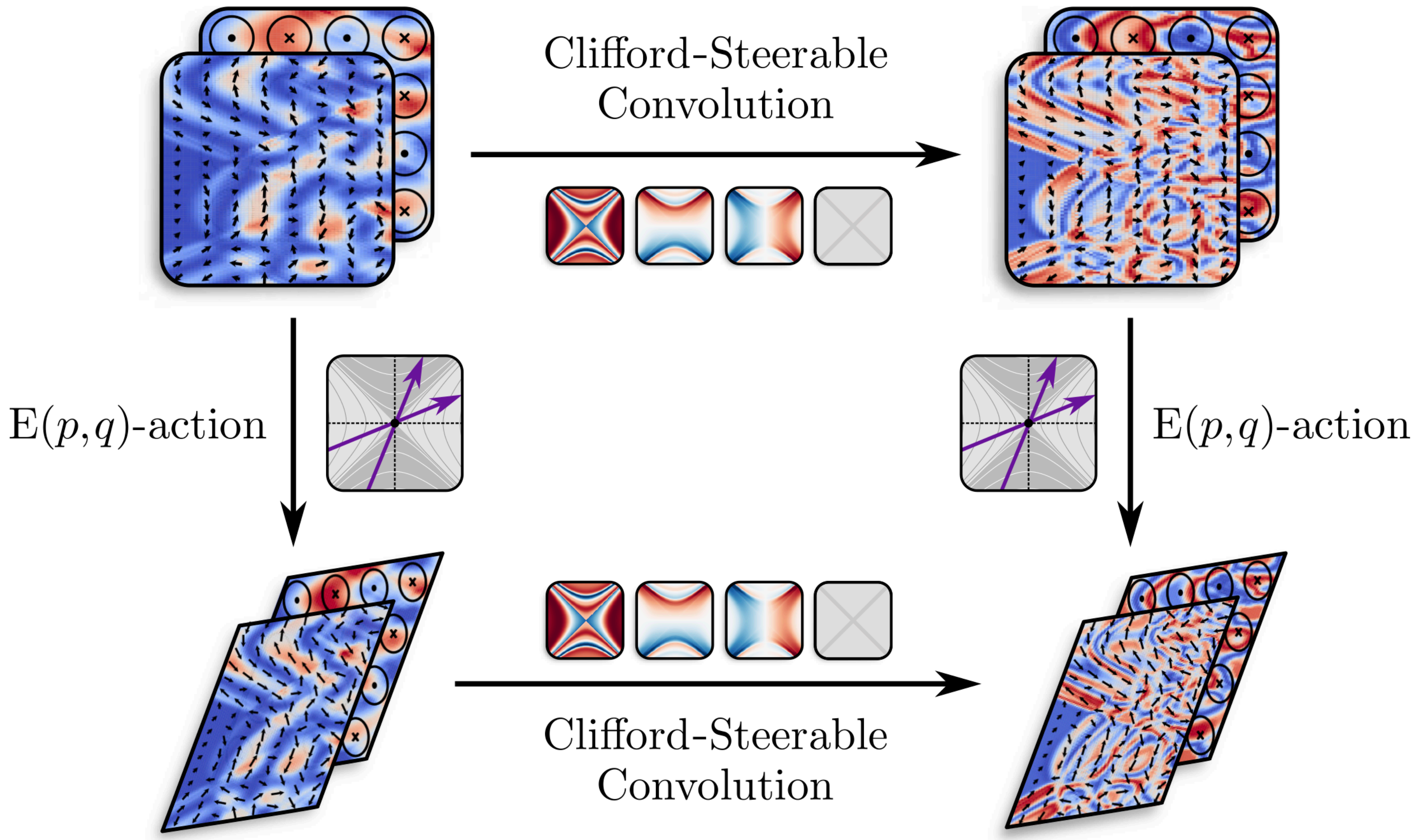
- Code is available at https://github.com/DavidRuhe/clifford-group-equivariant-neural-networks/
- Massive speed ups in JIT-compiled JAX versions.





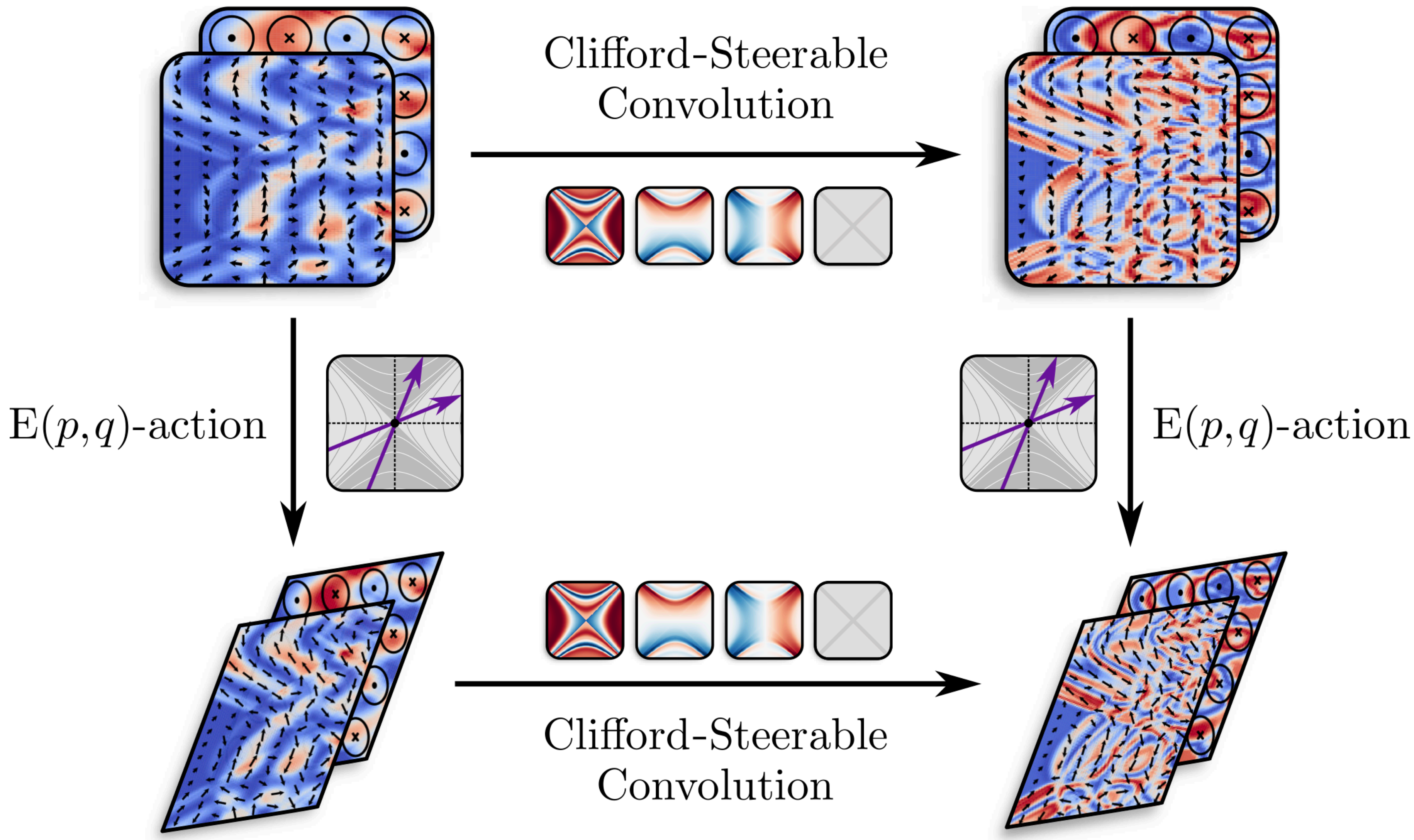
Adjacent & Followup Works

- Geometric Algebra Transformer (Brehmer et al., 2023, NeurIPS 2023)
  - Lorentz-Equivariant GATr (Spinner et al., 2024)
- Clifford Simplicial Message Passing (Liu et al., 2024, ICLR 2024)
- Clifford-Steerable CNNs (Zhdanov et al., 2024, ICML 2024)
- Applications in
  - 3D vision (Pepe et al., 2024)
  - (Bio)chemistry (Pepe et al., 2024)
  - Fluid Mechanics (Maruyana et al., 2024).



Adjacent & Followup Works

- Geometric Algebra Transformer (Brehmer et al., 2023, NeurIPS 2023)
    - Lorentz-Equivariant GATr (Spinner et al., 2024)
- **Clifford Simplicial Message Passing (Liu et al., 2024, ICLR 2024)**
- **Clifford-Steerable CNNs (Zhdanov et al., 2024, ICML 2024)**
- Applications in
    - 3D vision (Pepe et al., 2024)
    - (Bio)chemistry (Pepe et al., 2024)
    - Fluid Mechanics (Maruyana et al., 2024).

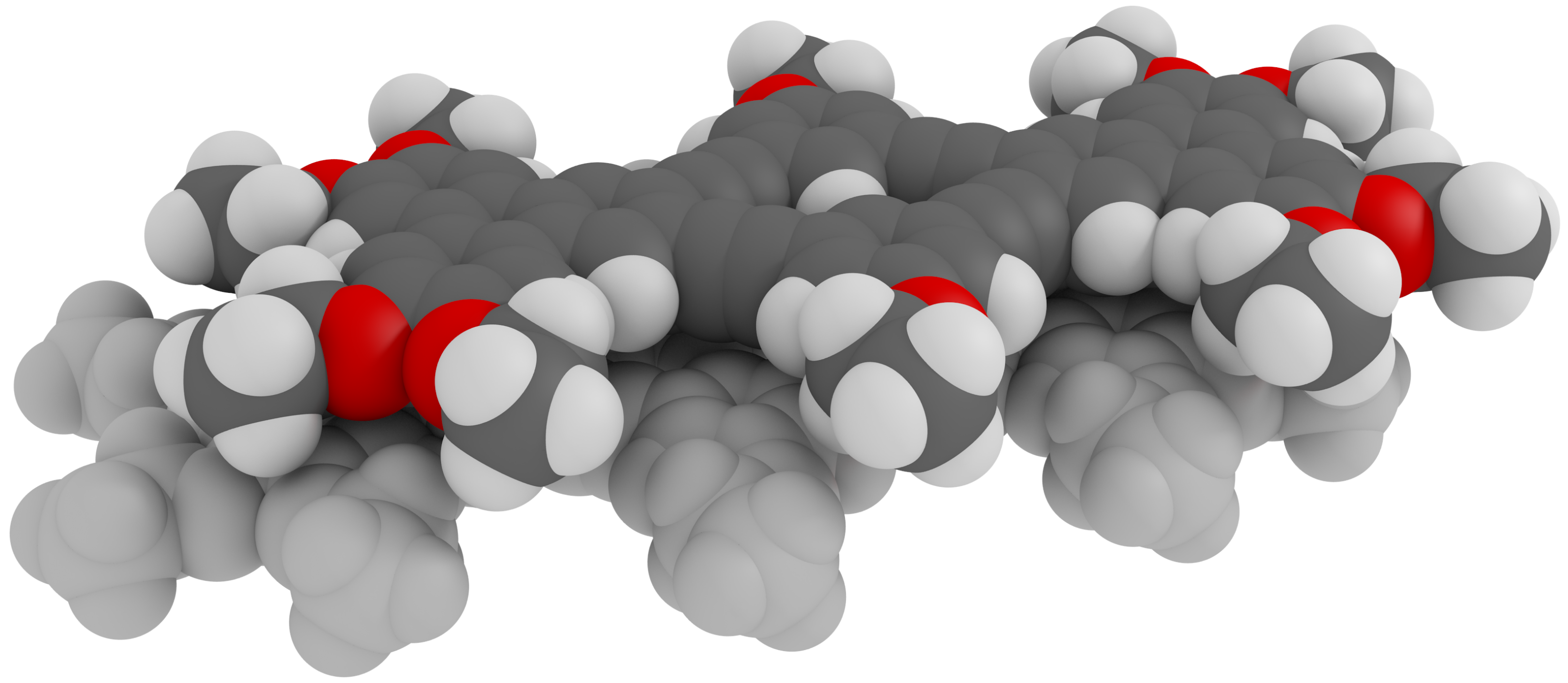


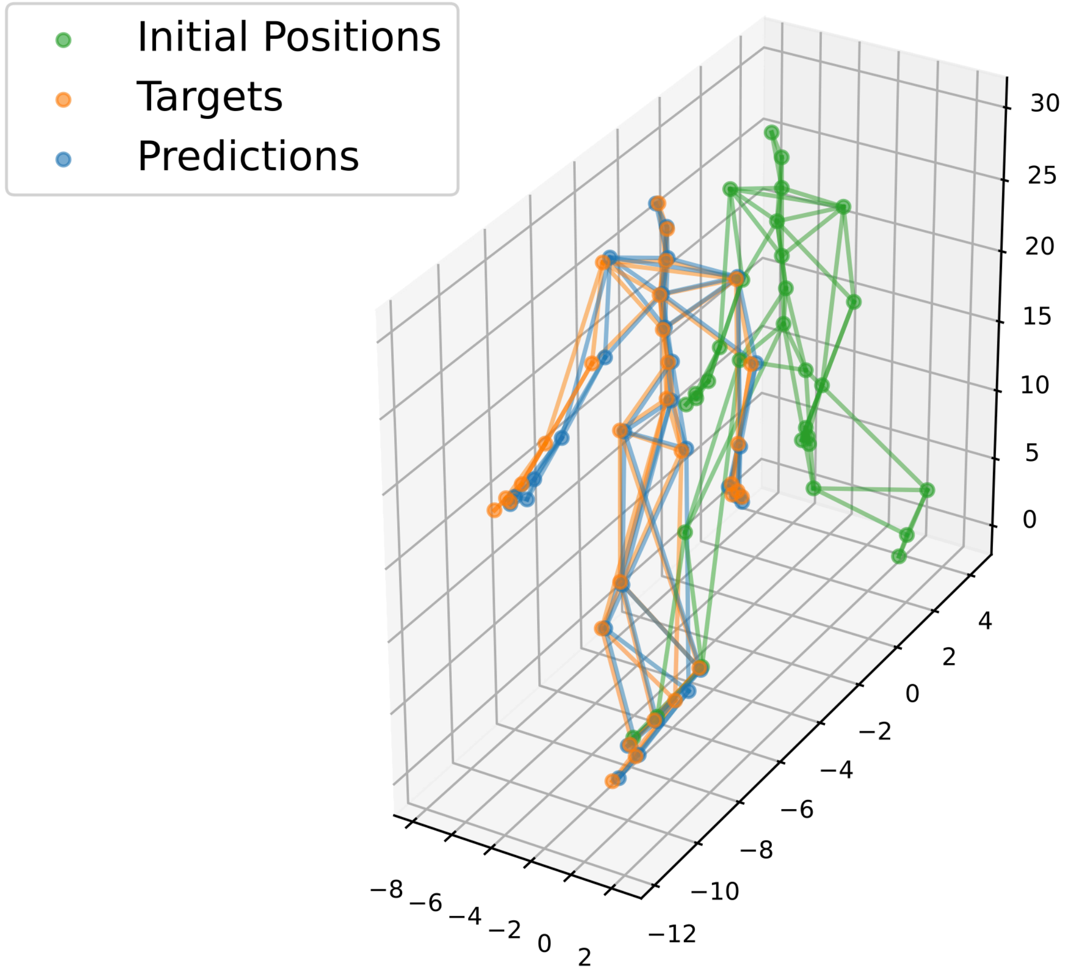
Clifford Group Equivariant Simplicial Message Passing Networks

Cong Liu[12,*], **David Ruhe**[123,*], **Floor Eijkelboom**[14], **Patrick Forré**[12]
AMLab, University of Amsterdam
{c.liu4,d.ruhe,f.eijkelboom,p.d.forre}@uva.nl

Motivation

- In data science, we can study e.g. molecules and proteins by equipping them with complex topologies.
- Graph Neural Networks are mostly used to tackle these challenges but they are only capable of modelling bi-interactions at each time.
- Can we find a method to both satisfy the equivariance constraint and being able model both geometries and topologies lie in the data?

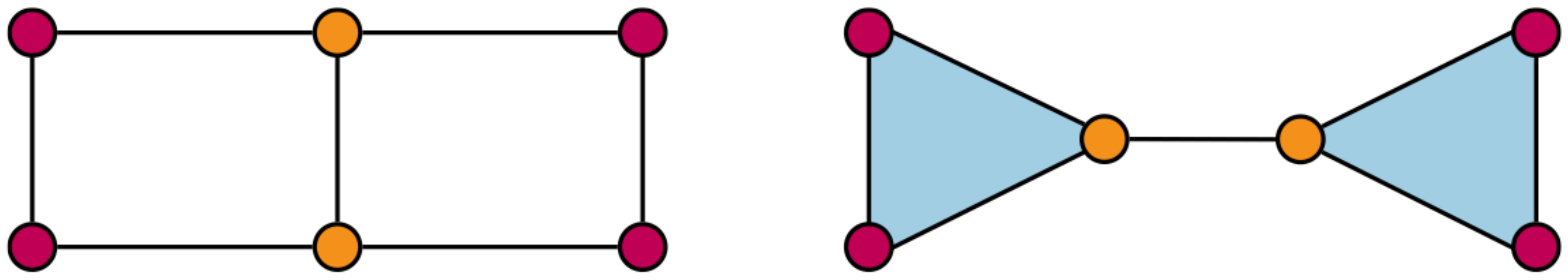




Message Passing Simplicial Networks

- Message Passing Networks are powerful, but they cannot distinguish two graphs with the same connectivity and the same set of nodes, even the two graphs have different topology.

- By lifting graphs to simplicial complex and pass messages on simplicial complex, we can identify them again!



- Message Passing Simplicial Networks learn the topological features in simplicial complex

Simplicial Complex

Definition 2.3 (Simplicial Complex)**.** *Let V be a finite set. An abstract simplicial complex K is a subset of the power set 2^V that satisfies:*

1. $\forall v \in V : \{v\} \in K$;
2. $\forall \sigma \in K : \forall \tau \subseteq \sigma, \tau \neq \emptyset : \tau \in K$.

- 0-simplex σ^0, nodes v_i
- 1-simplex σ^1, edges $\{v_i, v_j\}$
- 2-simplex σ^2, triangles $\{v_i, v_j, v_k\}$

Message Passing Simplicial Networks (MPSNs)

MPSN We propose a message passing model using the following message passing operations based on the four types of messages discussed in the previous section. For a simplex σ in a complex $\mathcal{K}$ we have:

E.g. edges of a triangle.

$$m_{\mathcal{B}}^{t+1}(\sigma) = \text{AGG}_{\tau \in \mathcal{B}(\sigma)}\Big(M_{\mathcal{B}}\big(h_\sigma^t, h_\tau^t\big)\Big) \quad (2)$$

E.g. triangles neighboring an edge.

$$m_{\mathcal{C}}^{t+1}(\sigma) = \text{AGG}_{\tau \in \mathcal{C}(\sigma)}\Big(M_{\mathcal{C}}\big(h_\sigma^t, h_\tau^t\big)\Big) \quad (3)$$

Other triangles that share an edge...

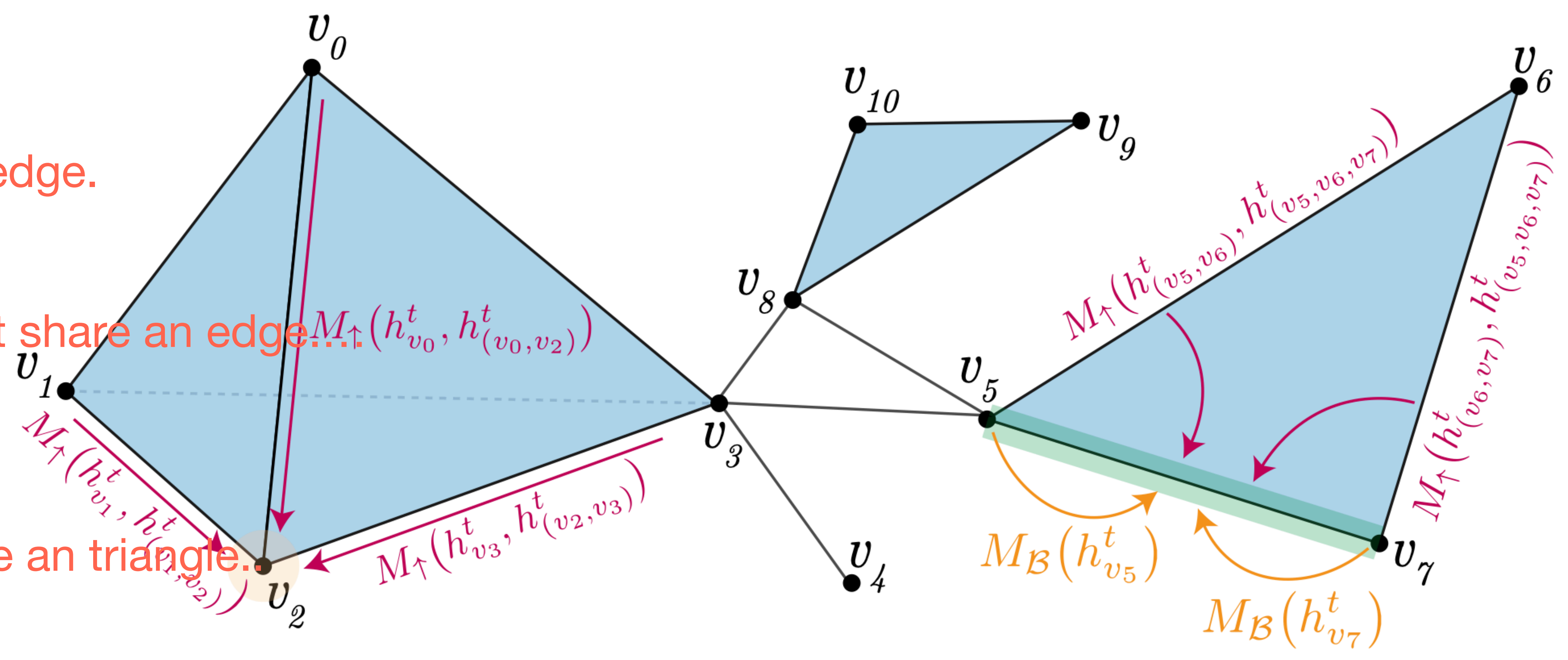
$$m_{\downarrow}^{t+1}(\sigma) = \text{AGG}_{\tau \in \mathcal{N}_\downarrow(\sigma)}\Big(M_{\downarrow}\big(h_\sigma^t, h_\tau^t, h_{\sigma \cap \tau}^t\big)\Big) \quad (4)$$

Other edges that share a triangle.

$$m_{\uparrow}^{t+1}(\sigma) = \text{AGG}_{\tau \in \mathcal{N}_\uparrow(\sigma)}\Big(M_{\uparrow}\big(h_\sigma^t, h_\tau^t, h_{\sigma \cup \tau}^t\big)\Big). \quad (5)$$

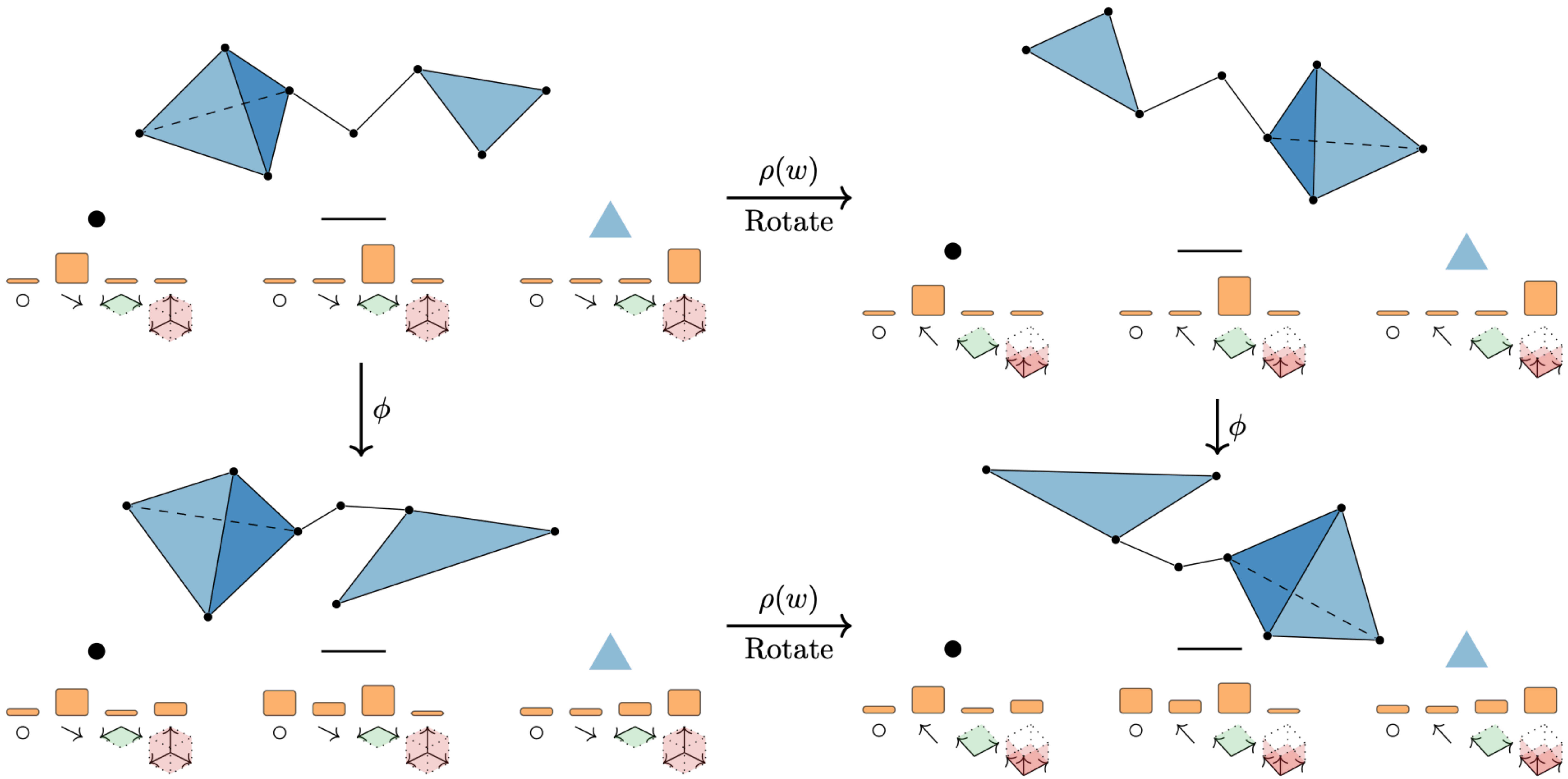
Then, the update operation takes into account these four types of incoming messages and the previous colour of the simplex:

$$h_\sigma^{t+1} = U\Big(h_\sigma^t, m_{\mathcal{B}}^t(\sigma), m_{\mathcal{C}}^t(\sigma), m_{\downarrow}^{t+1}(\sigma), m_{\uparrow}^{t+1}(\sigma)\Big). \quad (6)$$



(From Bodnar et al. 2021)

Clifford Group Equivariant Simplicial Message Passing

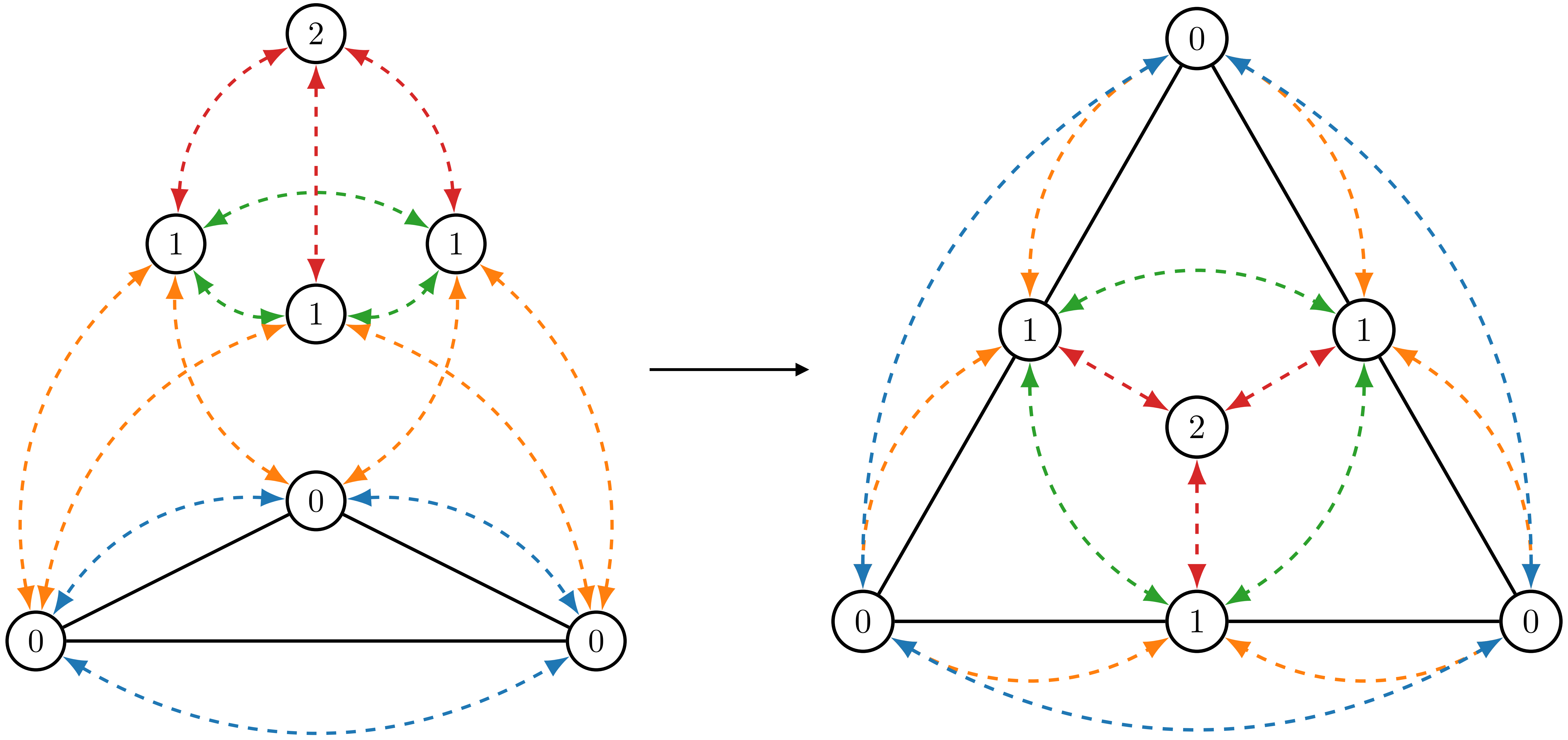


Shared Message Passing Networks

- In MSPNs, **every** type of communications between different dimensional simplices use different message networks.

- In this case, **6** networks are created and are forward propagated sequentially.

- We use only **1 shared** message passing network, *conditioned on communication type.*



Algorithm 1 Shared Simplicial Message Passing

Require: $K, \forall \sigma \in K : h^\sigma, \phi^m, \phi^h$

Repeat:

$$m^\sigma \leftarrow \operatorname{Agg}_{\substack{\tau \in B(\sigma) \\ \tau \in C(\sigma) \\ \tau \in N_\uparrow(\sigma) \\ \tau \in N_\downarrow(\sigma)}} \phi^m(h^\sigma, h^\tau, \dim \sigma, \dim \tau)$$

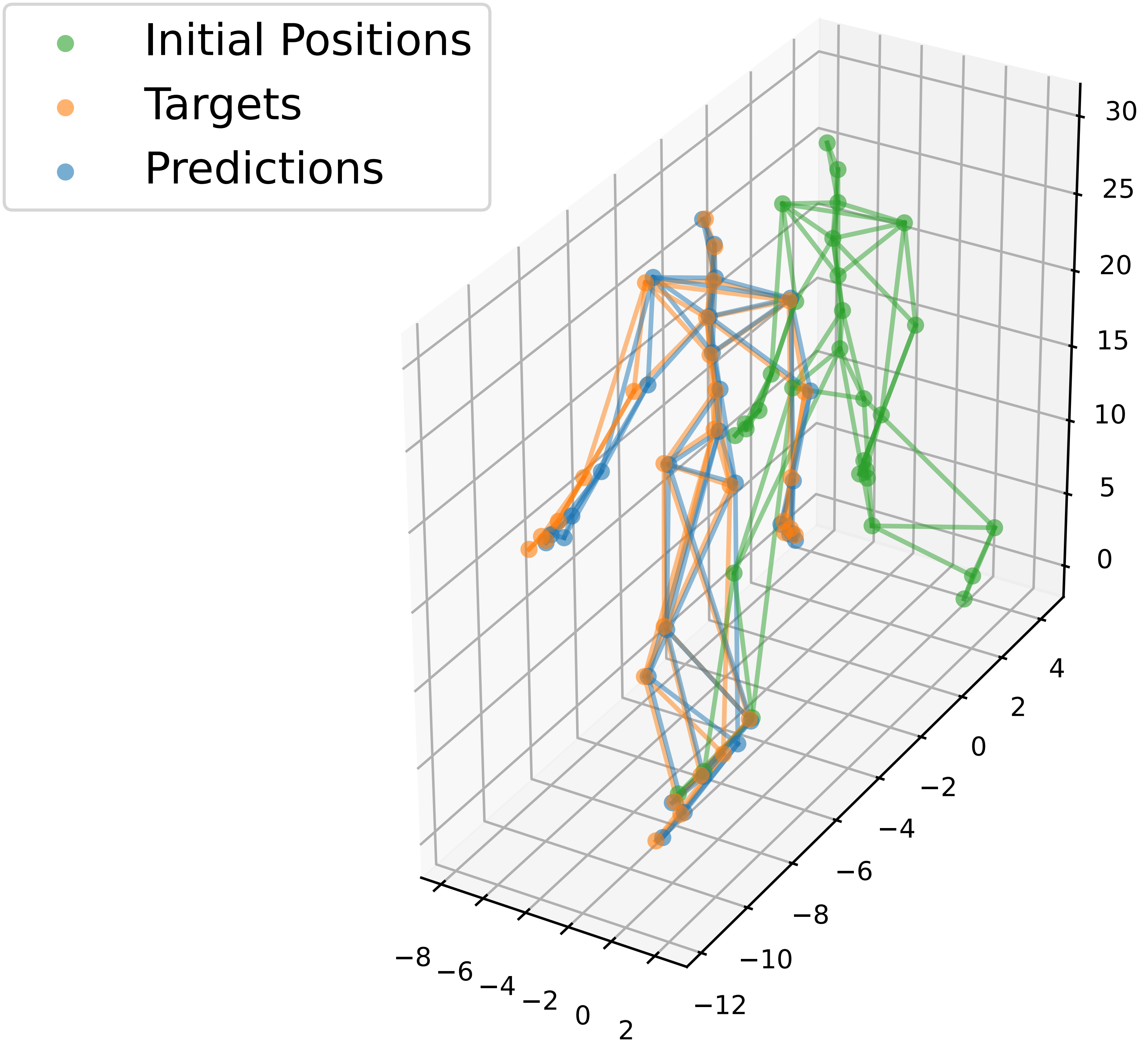
$$h^\sigma \leftarrow \phi^h \left(h^\sigma, m^\sigma, \dim \sigma\right)$$

Experiments

Human Walking Motion Prediction (E(2))

- Given 31 three-dimensional points coordinates , estimate the coordinates of these points after 30 time steps.

Method	MSE (↓)
Radial Field (Köhler et al., 2020)	197.0
TFN (Thomas et al., 2018)	66.9
SE(3)-Tr (Fuchs et al., 2020)	60.9
GNN (Gilmer et al., 2017)	67.3
EGNN (200K) (Satorras et al., 2021)	31.7
GMN (200K) (Huang et al., 2022)	17.7
EMPSN (200K)	15.1
CGENN (200K)	9.41
CSMPN (200K)	**7.55**



Experiments

MD17 Atomic Motion Prediction (E(3))

- Given the atomic positions at 10 separate time steps , estimate the coordinates of these atoms after serveral time steps.

	Aspirin	Benzene	Ethanol	Malonaldehyde
Radial Field (Köhler et al., 2020)	17.98 / 26.20	7.73 / 12.47	8.10 / 10.61	16.53 / 25.10
TFN (Thomas et al., 2018)	15.02 / 21.35	7.55 / 12.30	8.05 / 10.57	15.21 / 24.32
SE(3)-Tr (Fuchs et al., 2020)	15.70 / 22.39	7.62 / 12.50	8.05 / 10.86	15.44 / 24.47
EGNN (Satorras et al., 2021)	14.61 / 20.65	7.50 / 12.16	8.01 / 10.22	15.21 / 24.00
S-LSTM (Alahi et al., 2016)	13.12 / 18.14	3.06 / 3.52	7.23 / 9.85	11.93 / 18.43
NRI (Kipf et al., 2018)	12.60 / 18.50	1.89 / 2.58	6.69 / 8.78	12.79 / 19.86
NMMP (Hu et al., 2020)	10.41 / 14.67	2.21 / 3.33	6.17 / 7.86	9.50 / 14.89
GroupNet (Xu et al., 2022)	10.62 / 14.00	2.02 / 2.95	6.00 / 7.88	7.99 / 12.49
GMN-L (Huang et al., 2022)	9.76 / -	48.12 / -	4.83 / -	13.11 / -
EqMotion (300K) (Xu et al., 2023)	5.95 / 8.38	1.18 / 1.73	5.05 / 7.02	5.85 / 9.02
EMPSN (300K)	9.53 / 12.63	**1.03 / 1.12**	8.80 / 9.76	7.83 / 10.85
CGENN (300K)	**3.70 / 5.63**	**1.03** / 1.59	4.53 / 6.35	4.20 / 6.55
CSMPN (300K)	3.82 / 5.75	**1.03** / 1.60	**4.44 / 6.30**	**3.88 / 5.94**

Table 3: ADE / FDE (10^{-2}) (↓) of the tested models on the MD17 atomic motion dataset.

Experiments

NBA Players 2D Trajectory Prediction

- Given the player positions at 10 separate time steps , estimate the coordinates of these players for future 40 time steps.

	Attack	Defense
STGAT (Huang et al., 2019)	9.94 / 15.80	7.26 / 11.28
Social-Ways (Amirian et al., 2019)	9.91 / 15.19	7.31 / 10.21
Weak-Supervision (Zhan et al., 2019)	9.47 / 16.98	7.05 / 10.56
DAG-Net (200K) (Monti et al., 2020)	8.98 / 14.08	6.87 / 9.76
CGENN (200K)	9.17 / 14.51	6.64 / 9.42
CSMPN (200K)	**8.88 / 14.06**	**6.44 / 9.22**

Table 4: ADE / FDE (↓) of the tested models on the VUSport NBA player trajectory dataset.

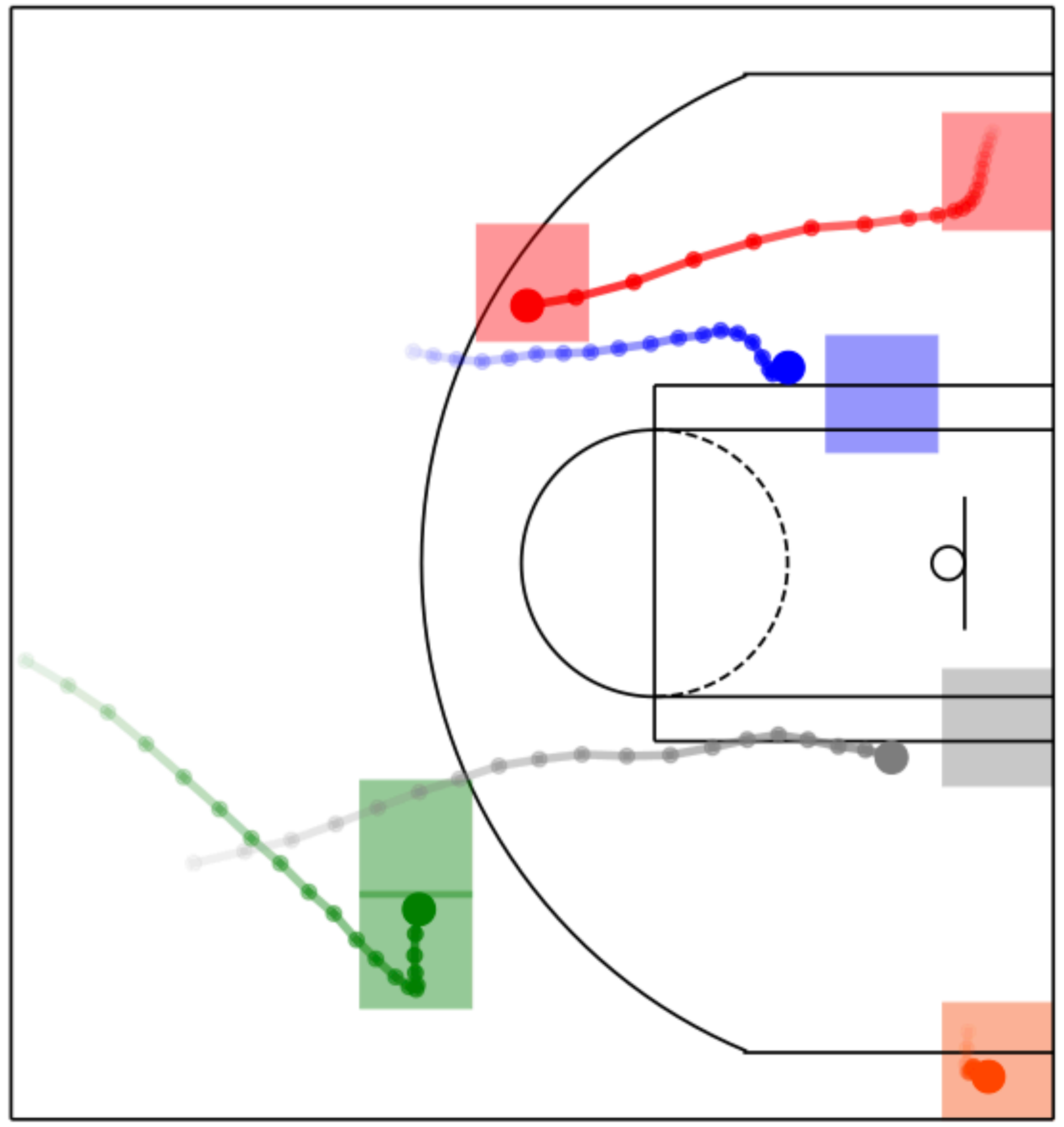


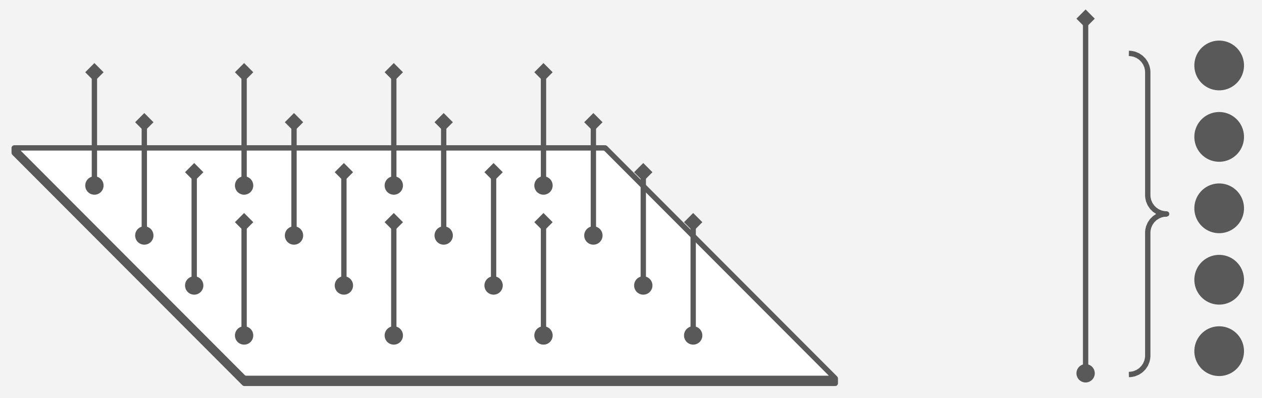
Figure from Alessio Monti, Alessia Bertugli, Simone Calderara, and Rita Cucchiara. Dag-net: Double attentive graph neural network for trajectory forecasting, 2020.

Clifford-Steerable Convolutional Neural Networks

Maksim Zhdanov[1] **David Ruhe**[*1 2 3] **Maurice Weiler**[*1] **Ana Lucic**[4] **Johannes Brandstetter**[5 6] **Patrick Forré**[1 2]

Feature Vector Fields

classic deep learning

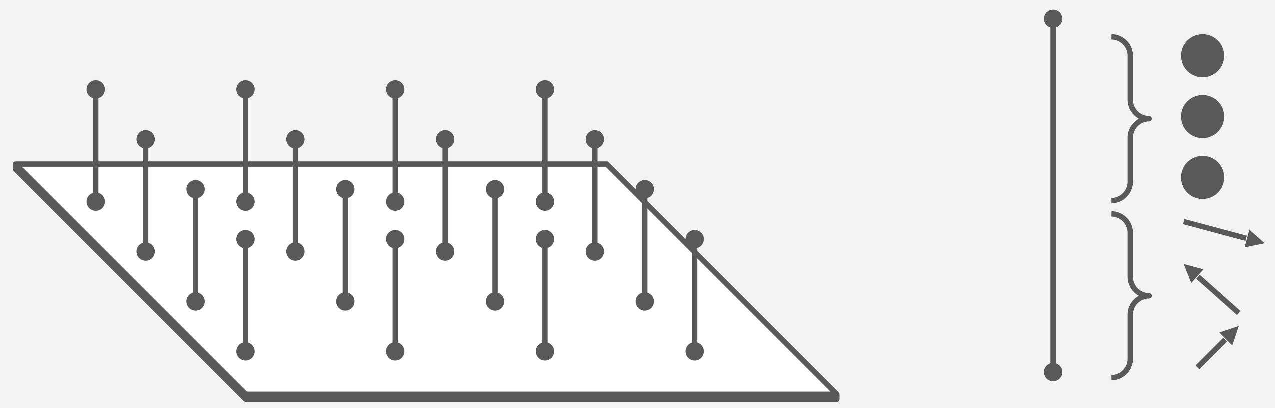


Euclidean space

scalars

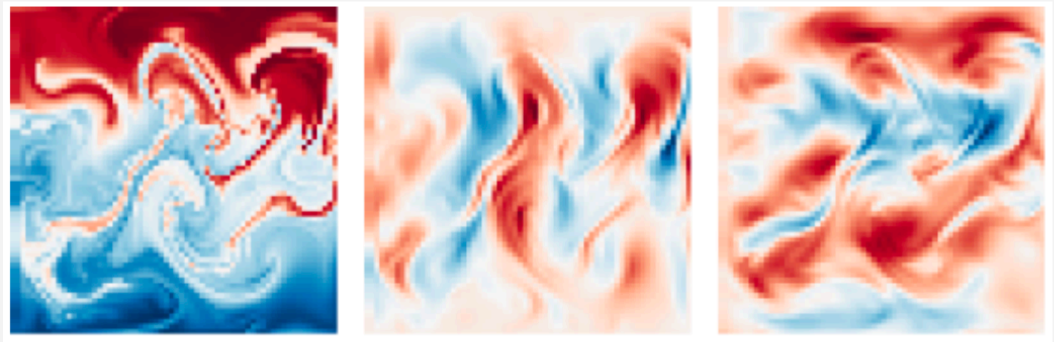
image data

geometric deep learning



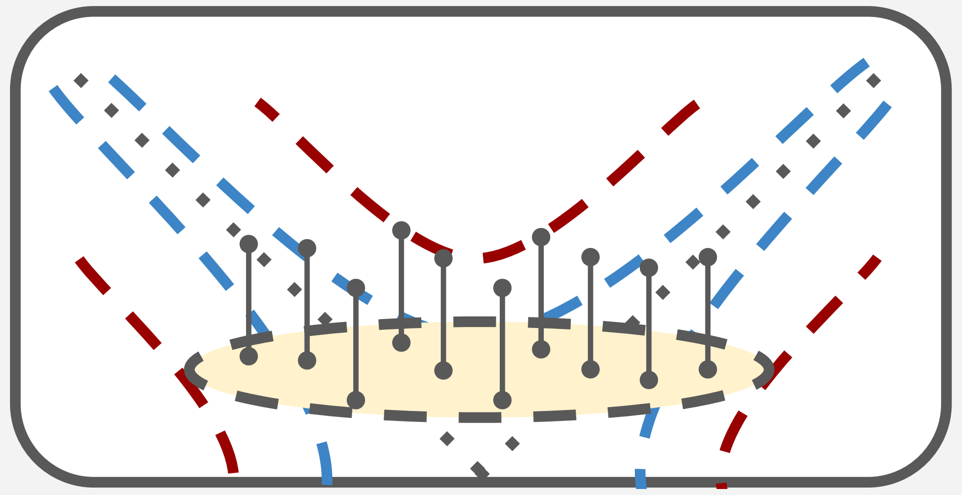
Euclidean space

tensors

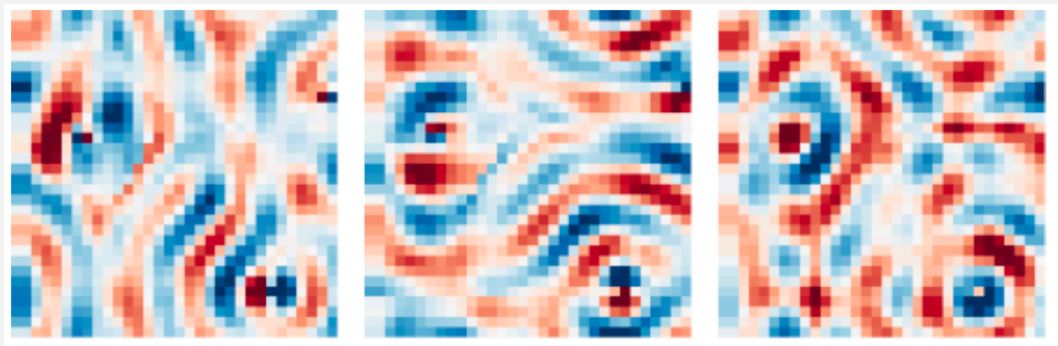


fluid dynamics data

what we need

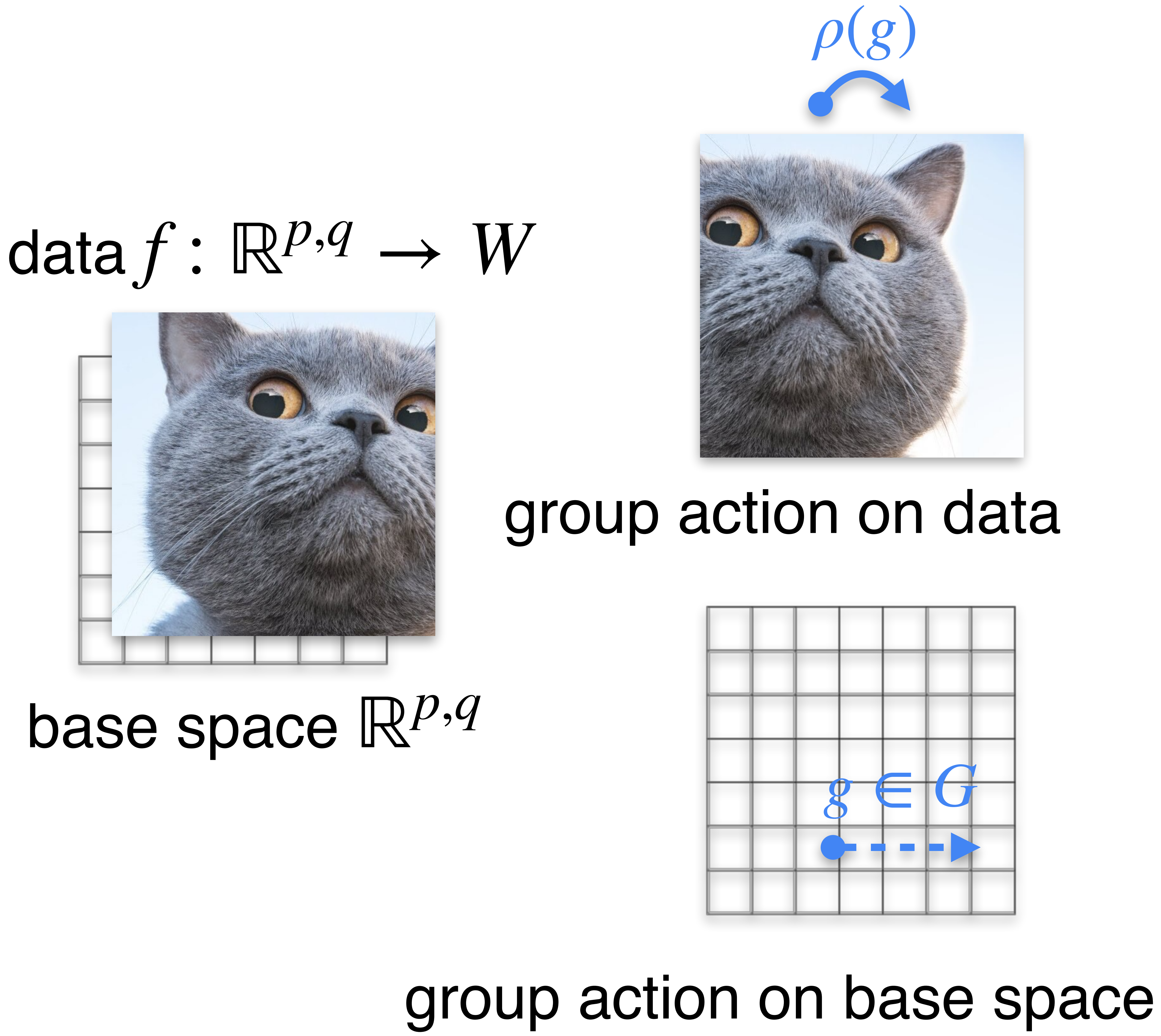


pseudo-Euclidean space



electromagnetic data

Data on Geometric Spaces

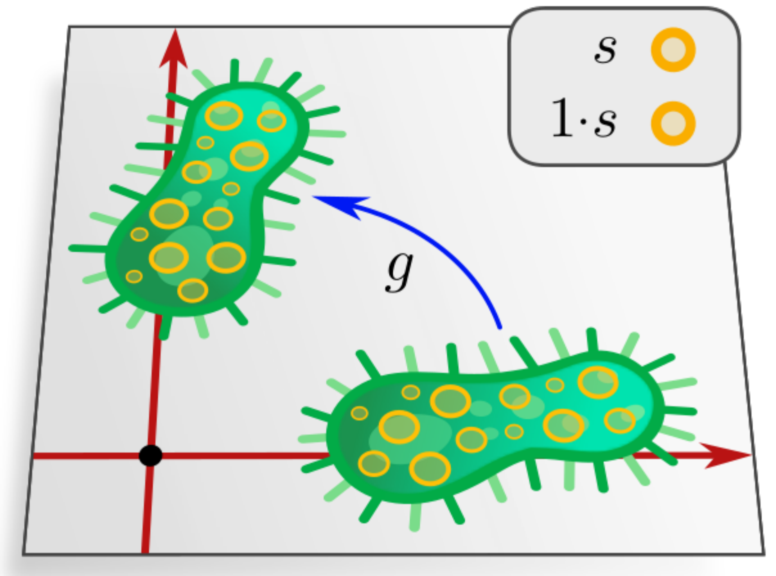


- ➔ transformations of the base space → transformations of the data.
- ➔ feature vector fields assign a feature $f(x)$ to each point $x \in \mathbb{R}^{p,q}$:

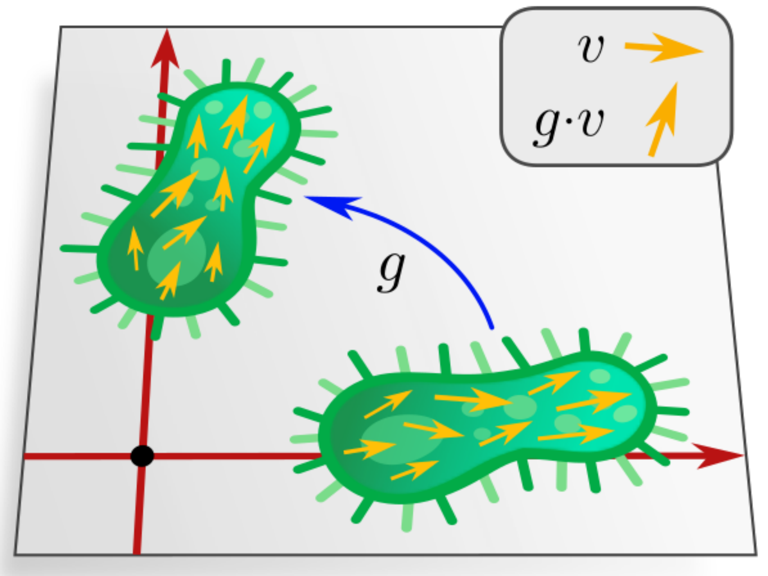
$$f: \mathbb{R}^{p,q} \to W$$

- ➔ feature fields are equipped with transformation rules under group actions g - representations $\rho(g)$.

Data on Geometric Spaces



scalar field, $\rho(g) = 1$



vector field, $\rho(g) = g$

different types of feature fields

- ➔ transformations of the base space → transformations of the data.
- ➔ feature vector fields assign a feature $f(x)$ to each point $x \in \mathbb{R}^{p,q}$:

$$f : \mathbb{R}^{p,q} \to W$$

- ➔ feature fields are equipped with transformation rules under group actions g - representations $\rho(g)$.

source: https://github.com/QUVA-Lab/escnn

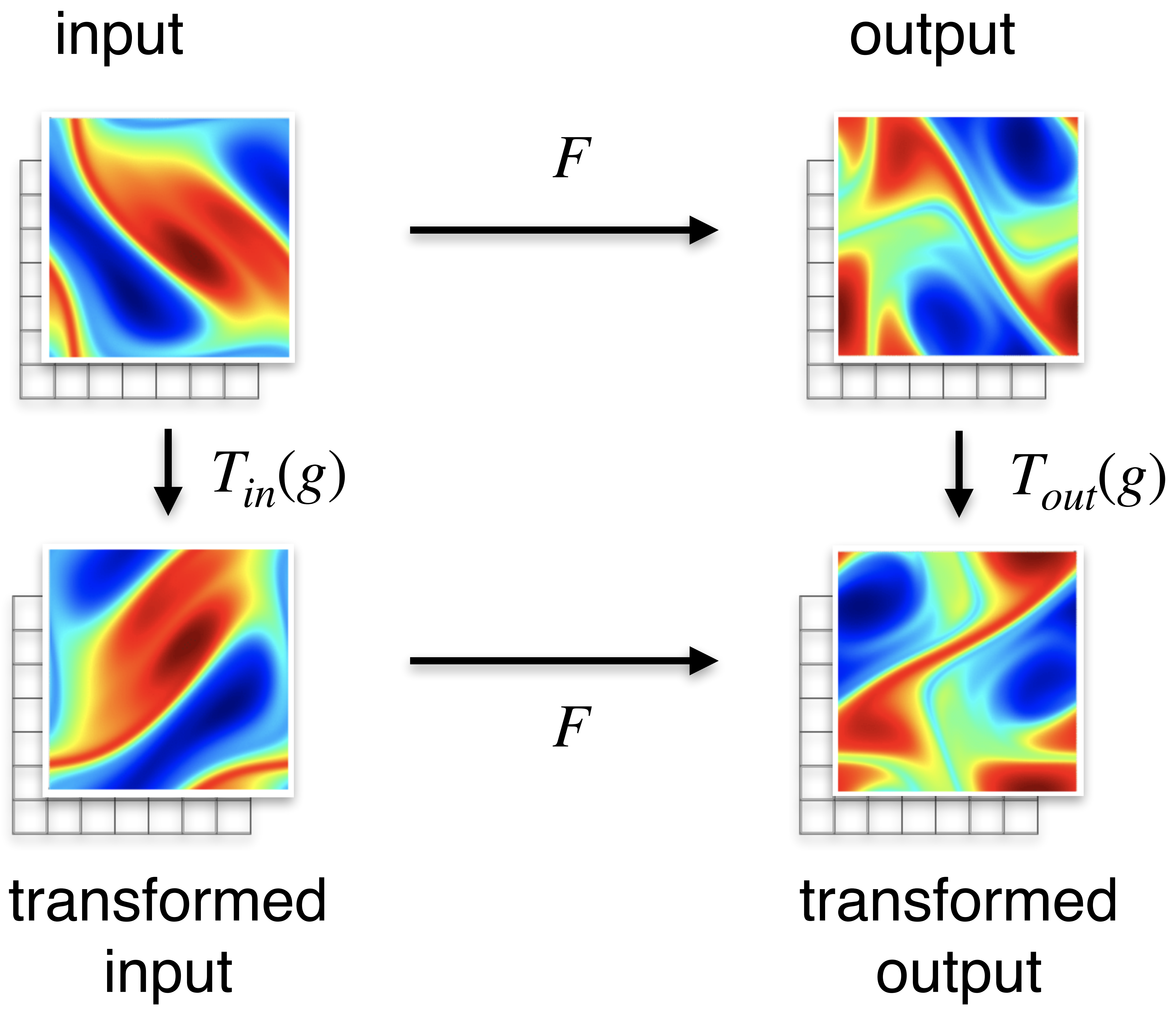
Functions on Geometric Spaces

- our goal is to approximate the map between two feature spaces:

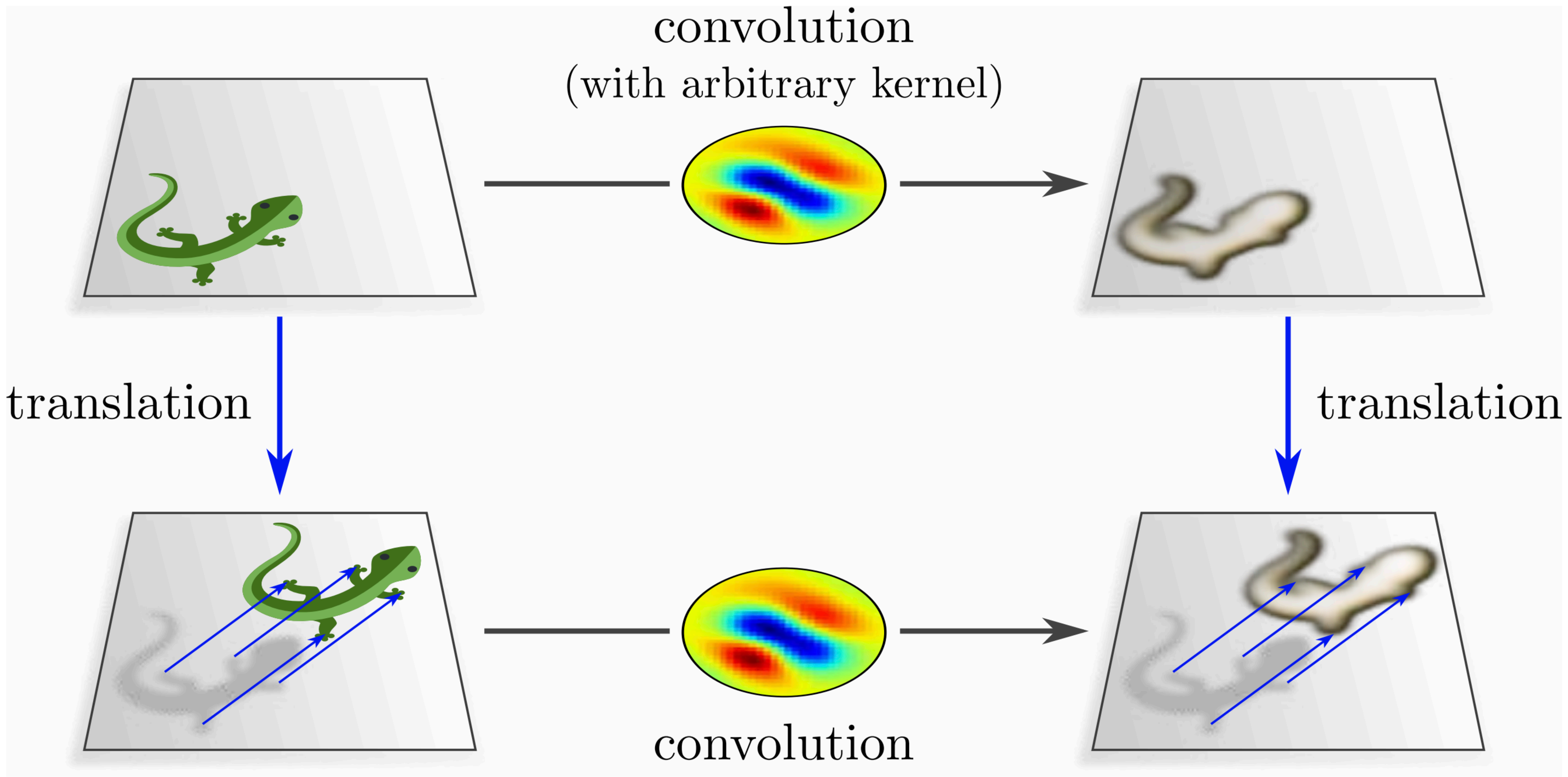
$$F : f_{in} \rightarrow f_{out}$$

- since every feature field is equipped with its group representation, the map must respect it = equivariant:

$$F \circ \rho_{in}(g) = \rho_{out}(g) \circ F$$



Convolutional Neural Networks

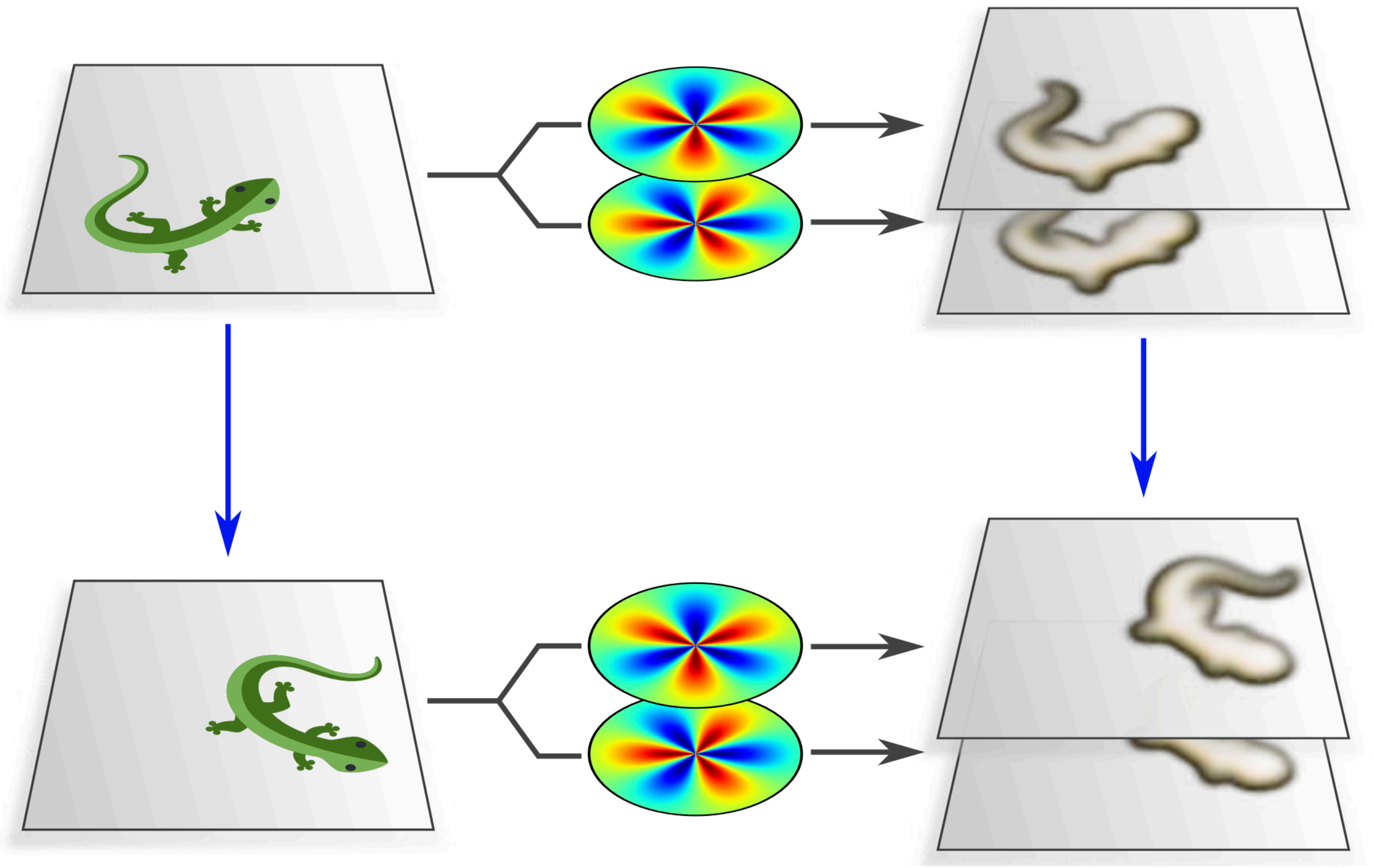


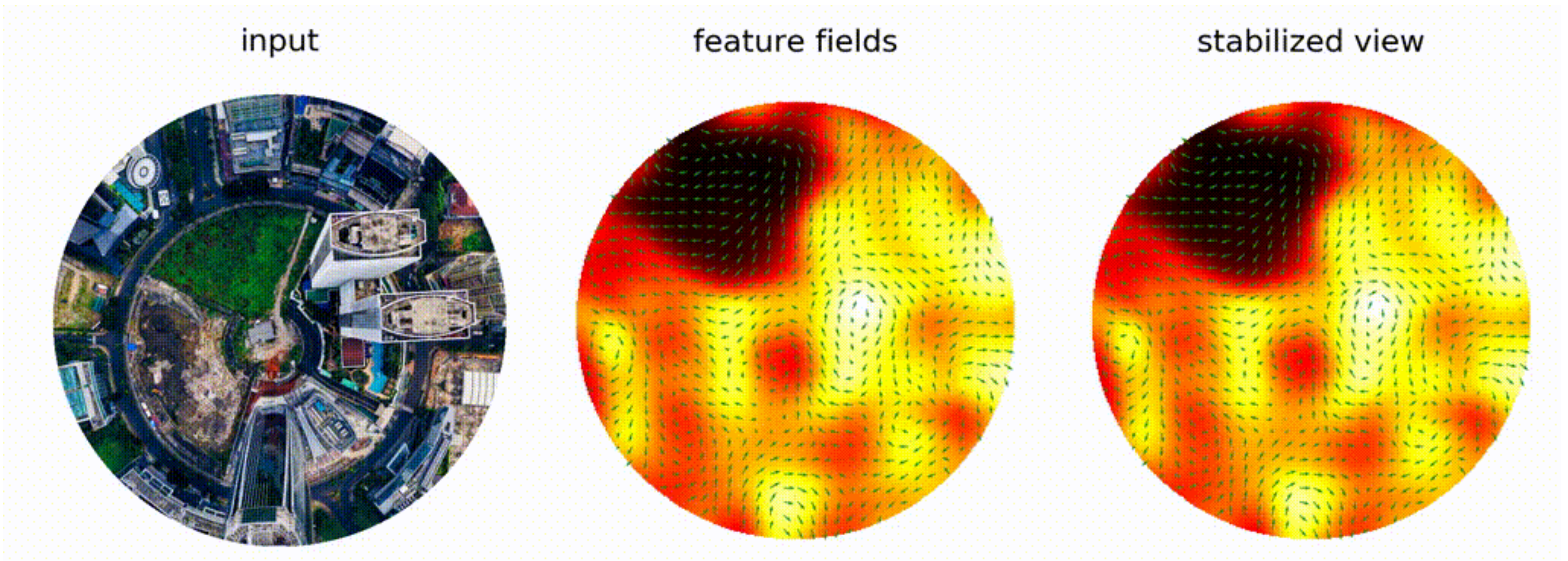
➔ convolutional layer:

$$(f_{in} * k)(x) = \int_{-\infty}^{\infty} f_{in}(\tau)k(x-\tau)d\tau$$

➔ it is translation-equivariant → pattern recognition power.

Steerable CNNs





- for arbitrary group G, one can put a constraint on kernels:

$$k(g\,.\,x) = \rho_{\text{out}}(g)k(x)\rho_{\text{in}}(g)^T \qquad \forall g \in G$$

- guarantees G-equivariance of a convolutional layer.
- Zhdanov et al., 2023 show that this can be solved **implicitly**.

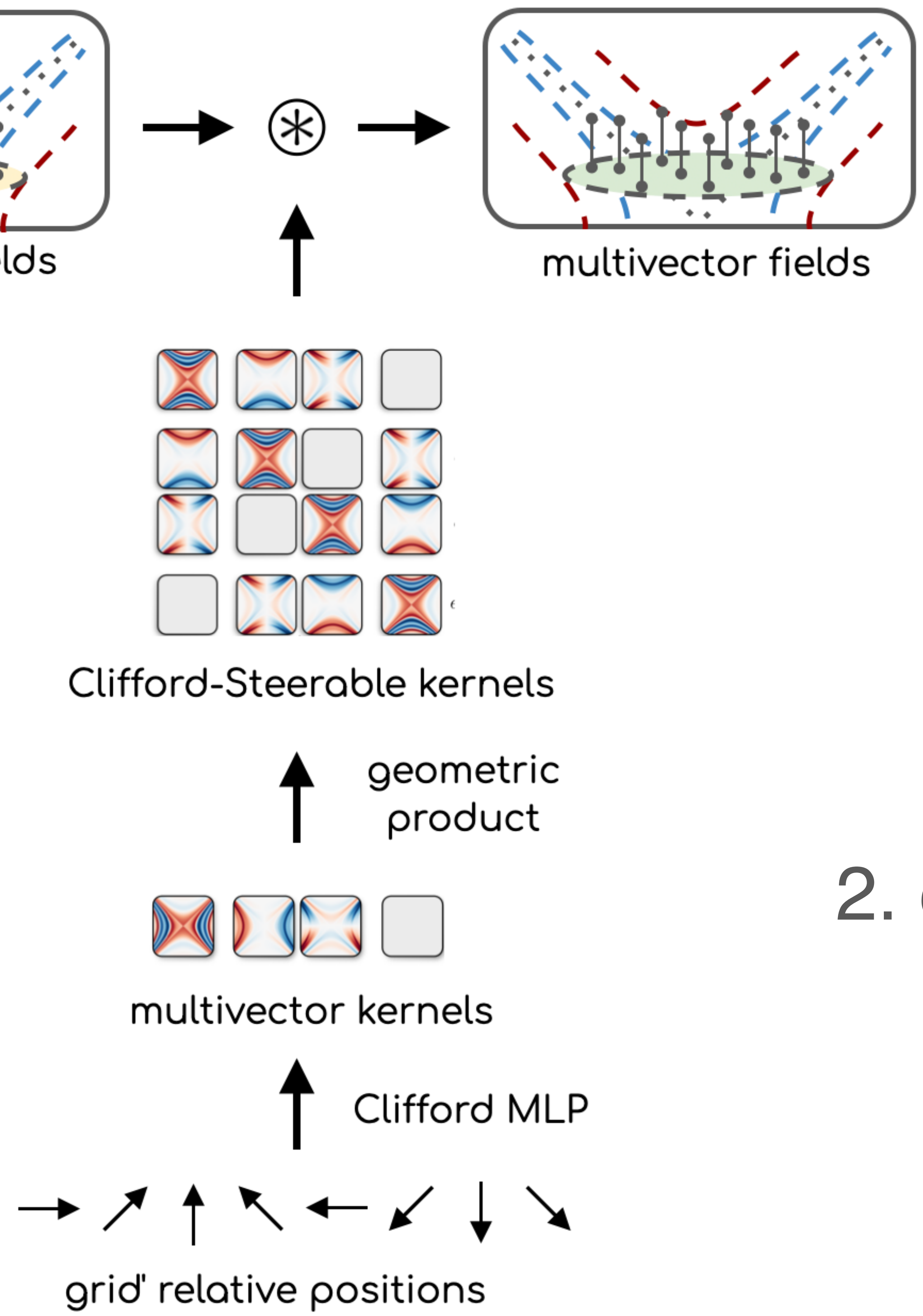
Clifford-Steerable Implicit Kernels

4. compute convolution

3. partially evaluate geometric product

2. compute kernel matrix

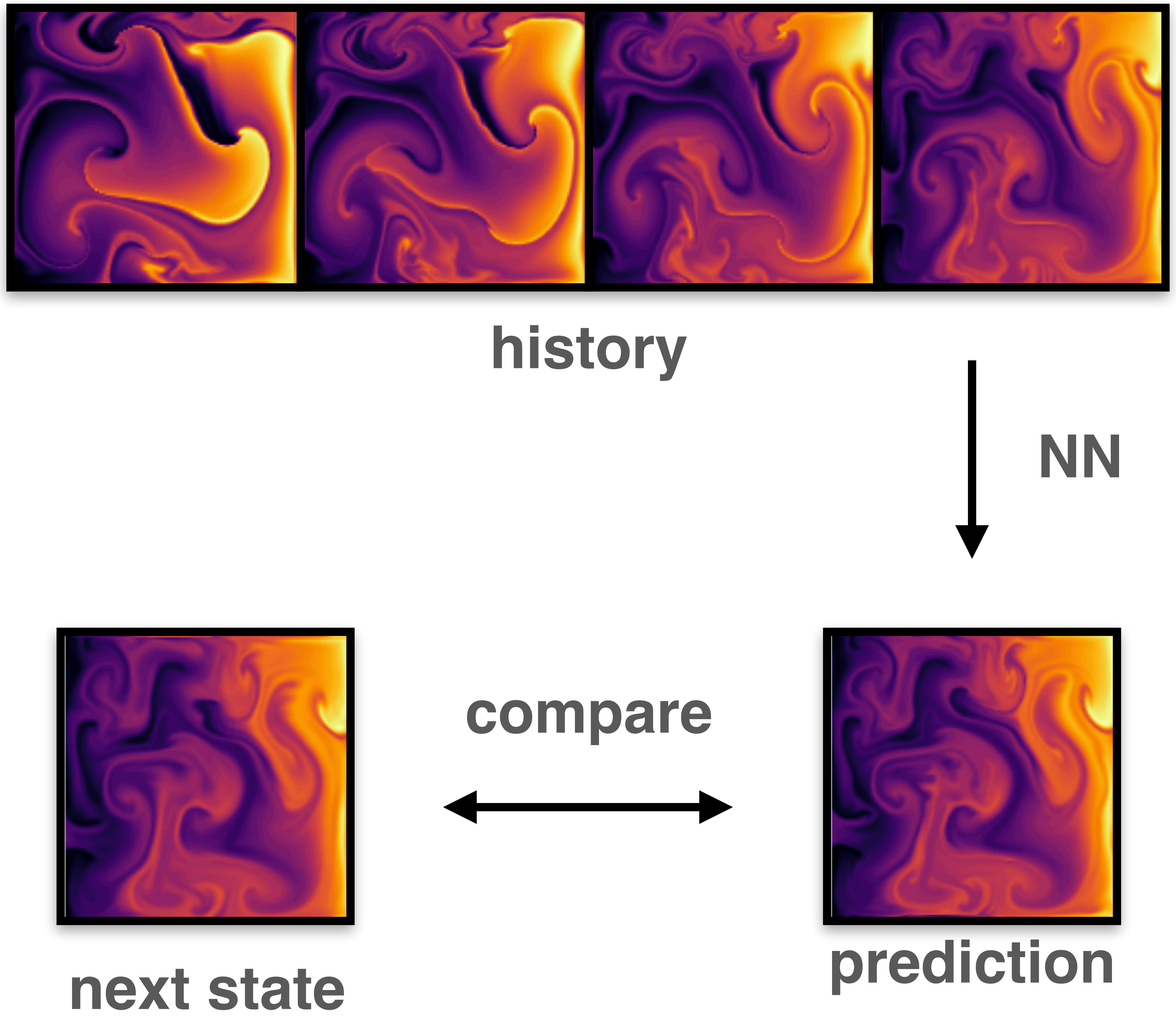
1. define kernel grid (e.g. 3x3)



Experiments

- ➔ in every experiment, the task is to predict a future state given the history.
- ➔ for classical physics, each time step is a separate image.
- ➔ for relativistic physics, time is part of the grid (aka video).

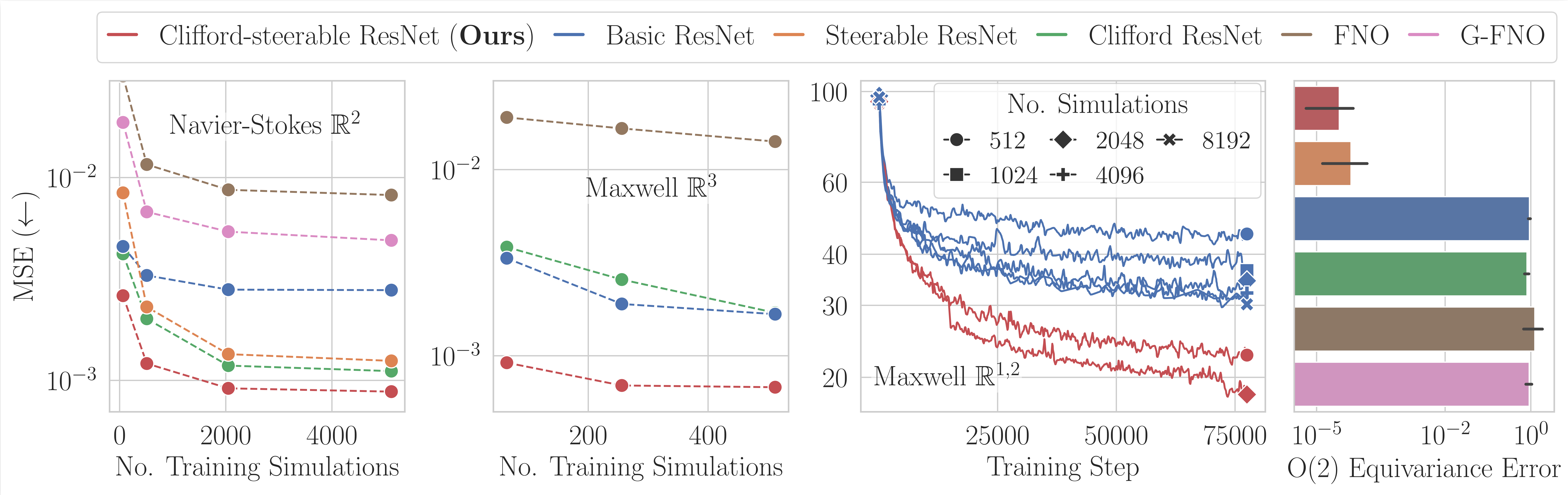
example: fluid dynamics



Experiments

we compare the framework against multiple (equiv-t) convolutional operators:

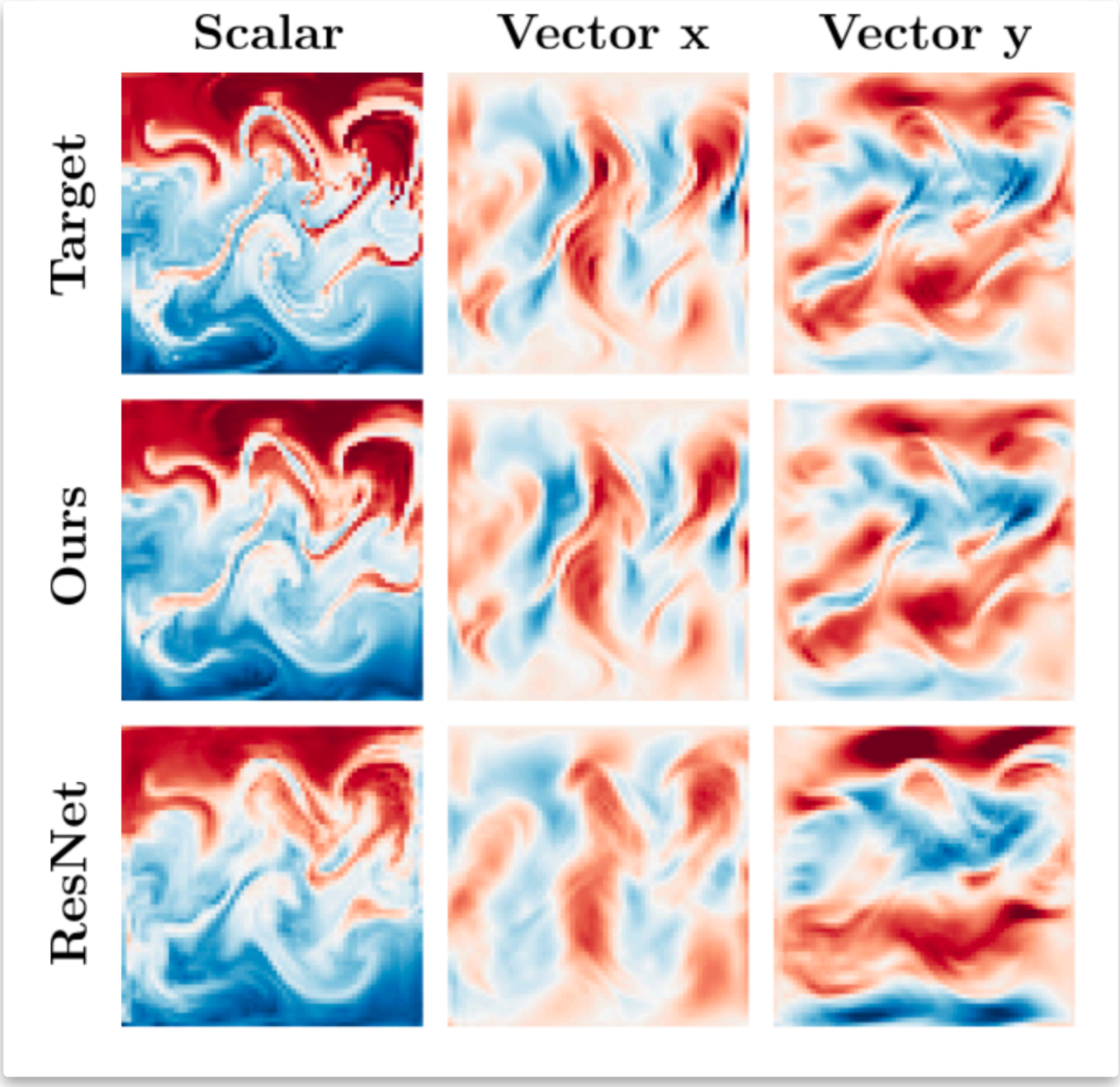
fluid dynamics **electrodynamics** **relativistic electrodynamics** **equivariance error**



Fluid Mechanics

equivariance allows for out-of-distribution generalizability across isometries:

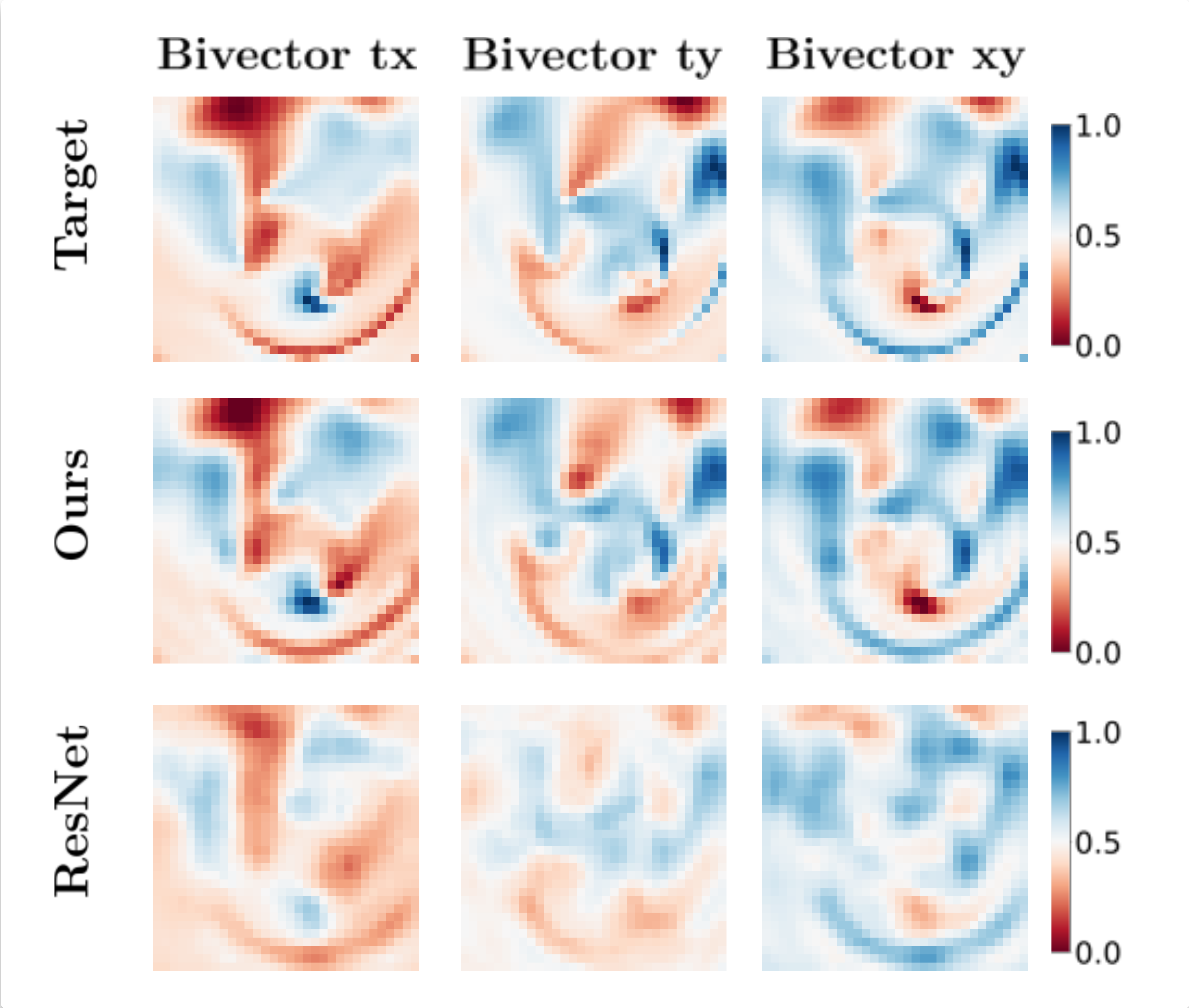
trained on
64 trajectories



trained on 5120
trajectories

Electrodynamics

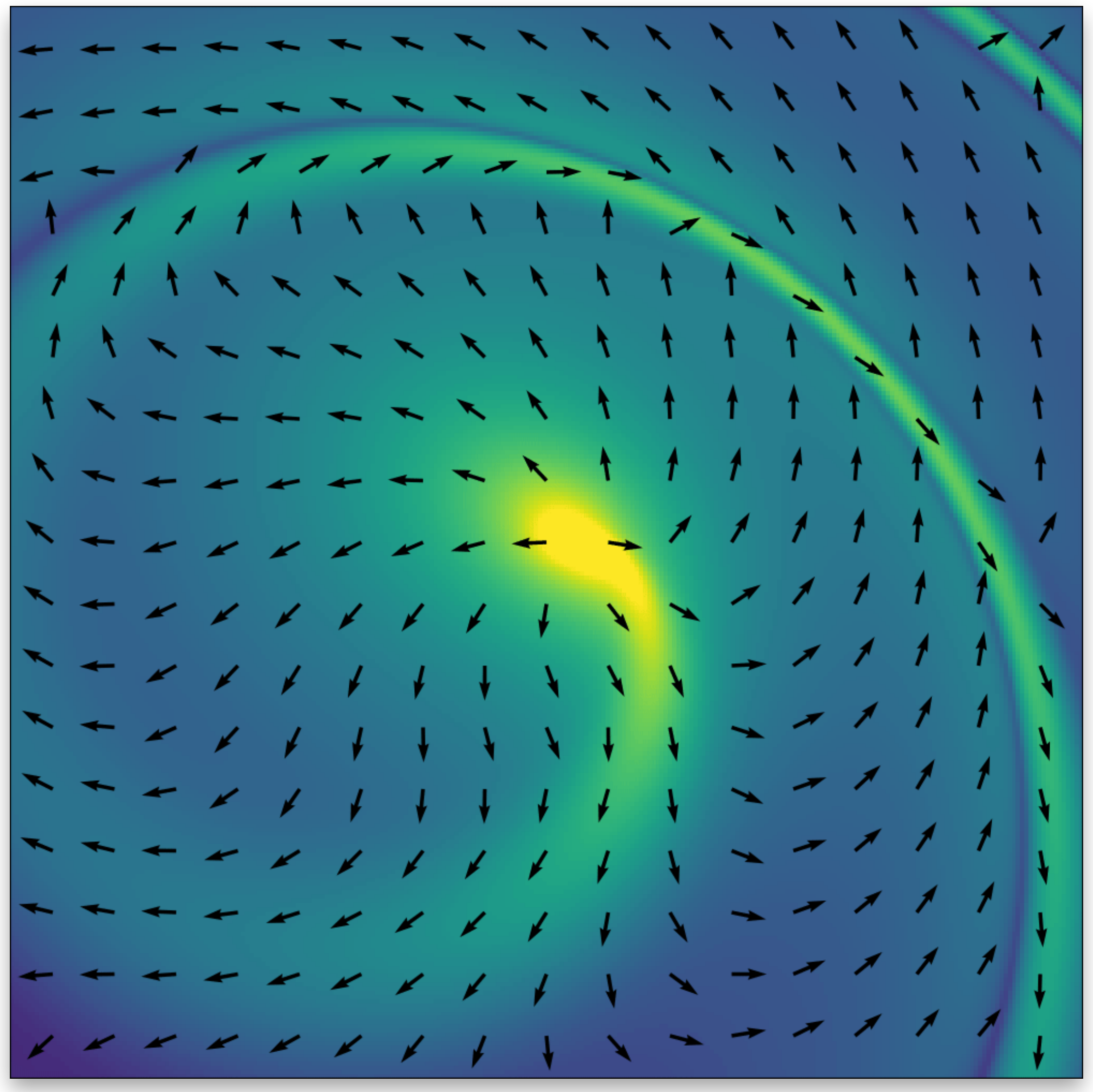
equivariance allows for out-of-distribution generalizability across isometries:



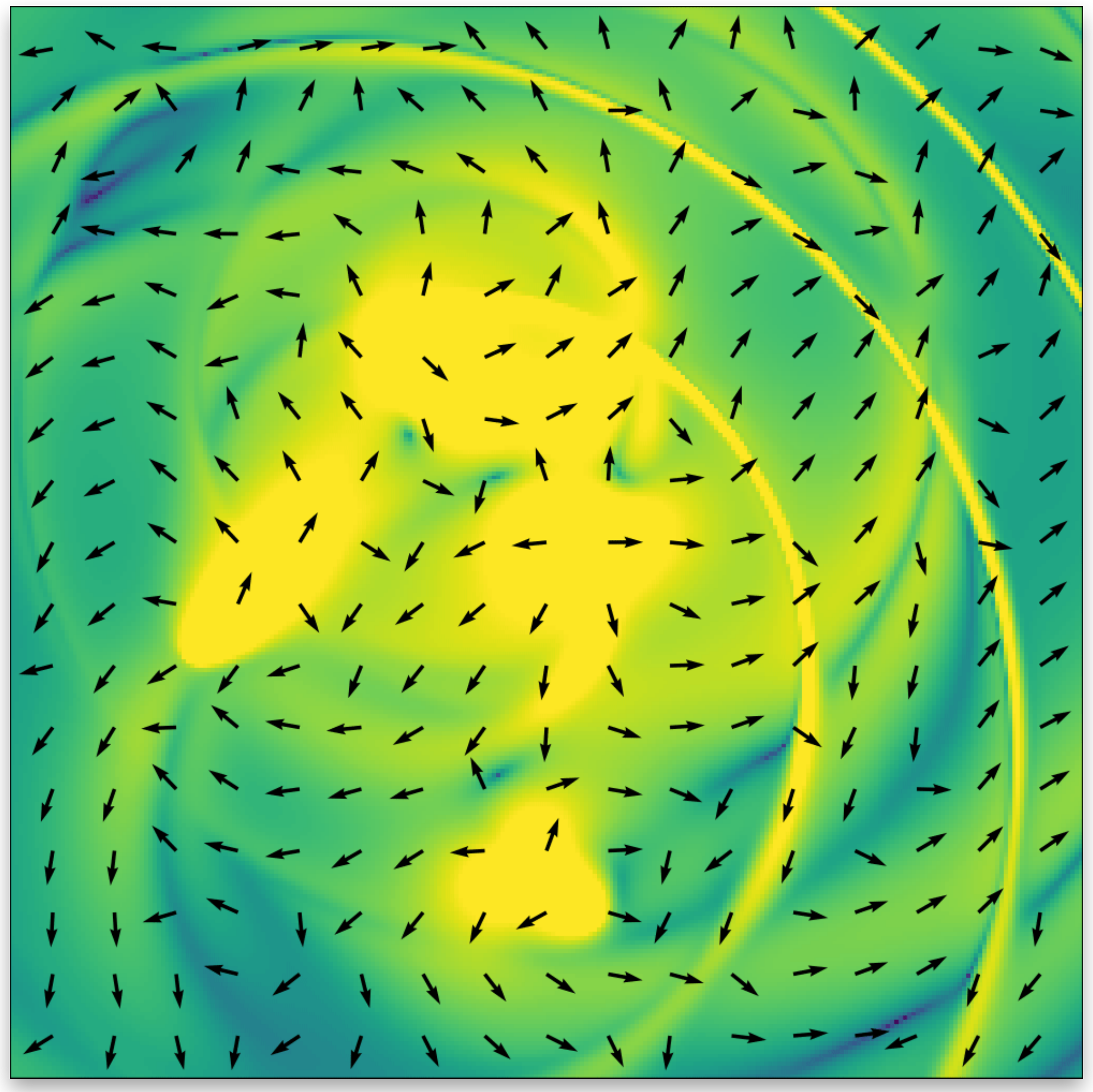
CSCNNs capture crisper details

Relativistic Electrodynamics

data: EM fields are emitted by point sources that move, orbit and oscillate at relativistic speeds.



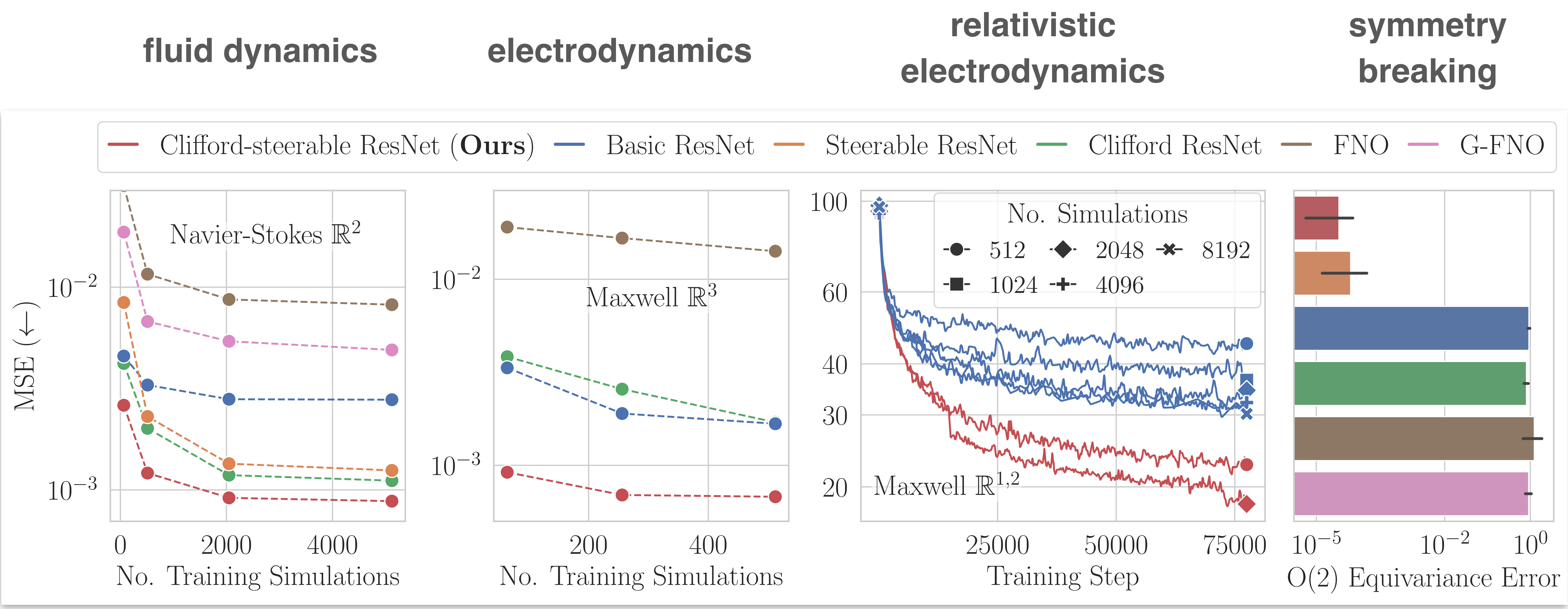
1 charge



5 charges

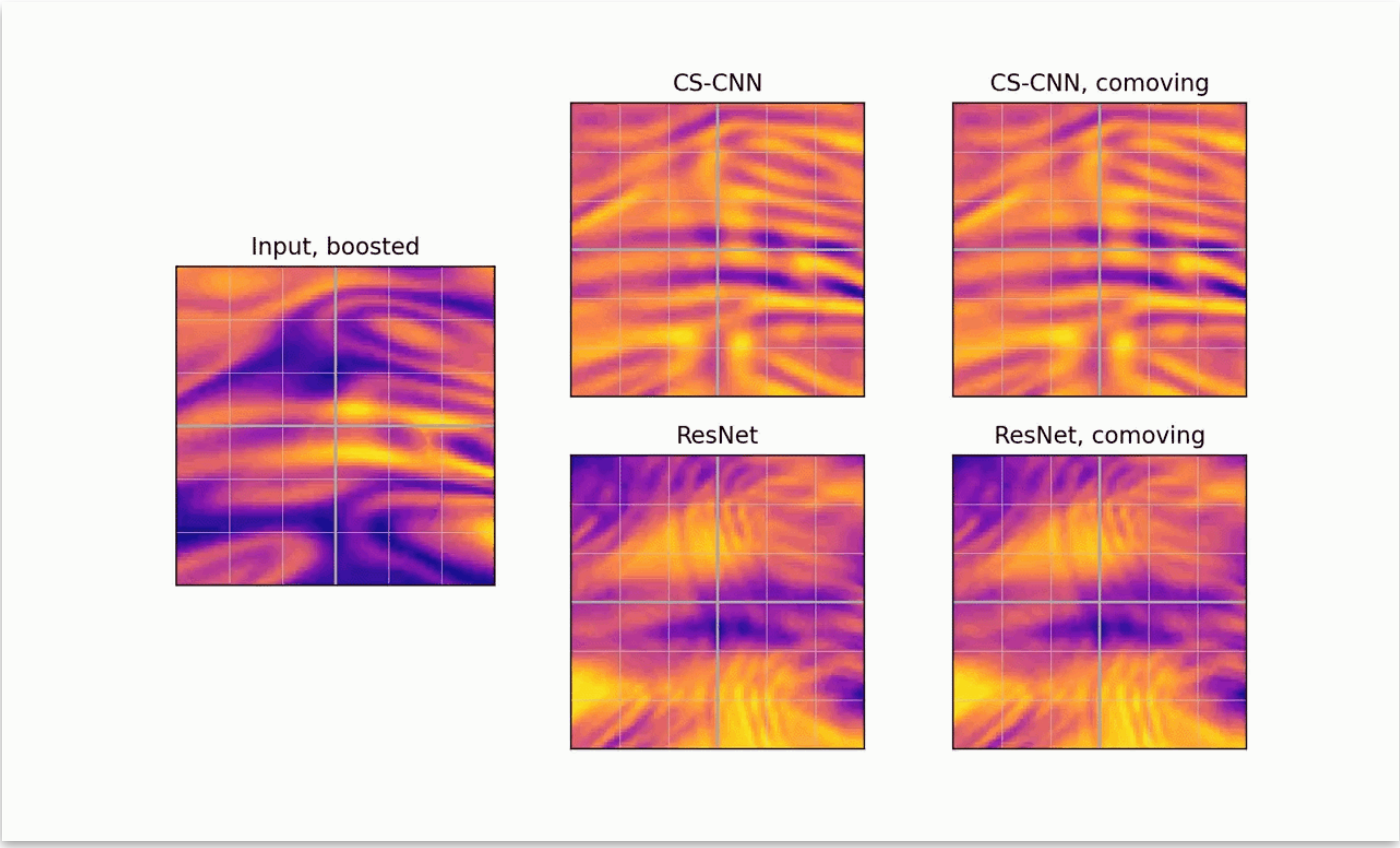
Experiments

we compare the framework against multiple (equiv-t) convolutional operators:



Experiments

we are now able to implement Lorentz-equivariant CNNs, e.g. equivariant to Lorentz boosts:



Thanks

Please contact me at **david.ruhe@gmail.com** **!**