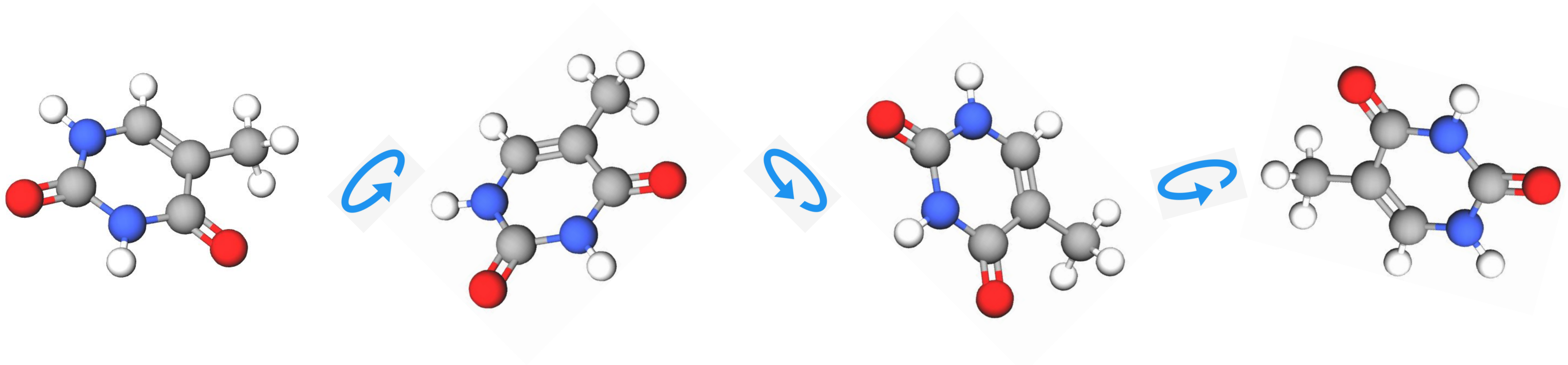


# Incorporating symmetry into deep learning models

## FFT seminar, 08.12.2022


Maksim Zhdanov

HZDR · FWKT · Maksim Zhdanov · m.zhdanov@hzdr.de · www.hzdr.de

# Motivation

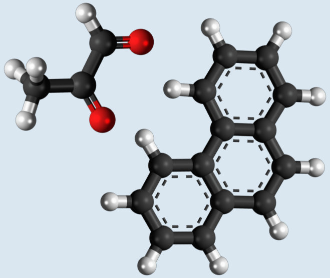
## Modeling real-world phenomena

**Image classification**

define the content of
an image

https://dl.acm.org/doi/10.1145/3065386

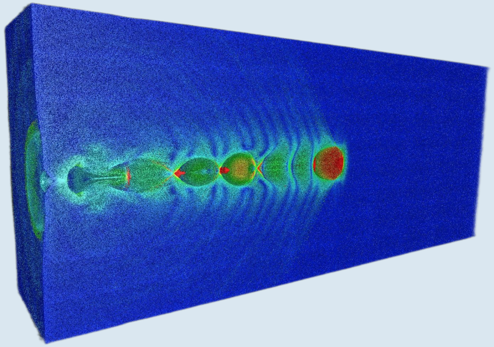
**Molecular chemistry**



simulations of
molecules

https://arxiv.org/abs/2105.09016

**Physical processes**

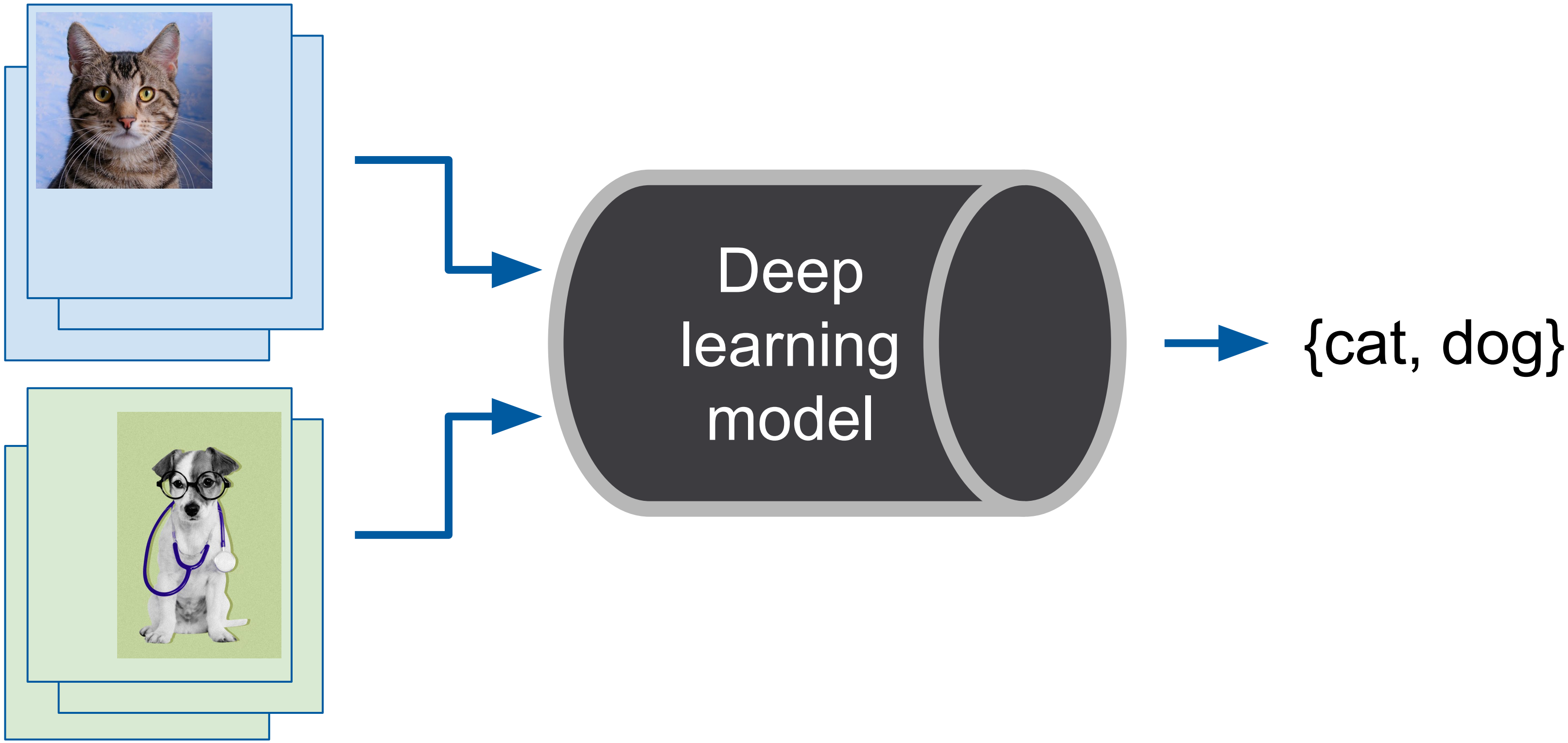


simulating laser -
plasma interaction

https://ieeexplore.ieee.org/document/5556015

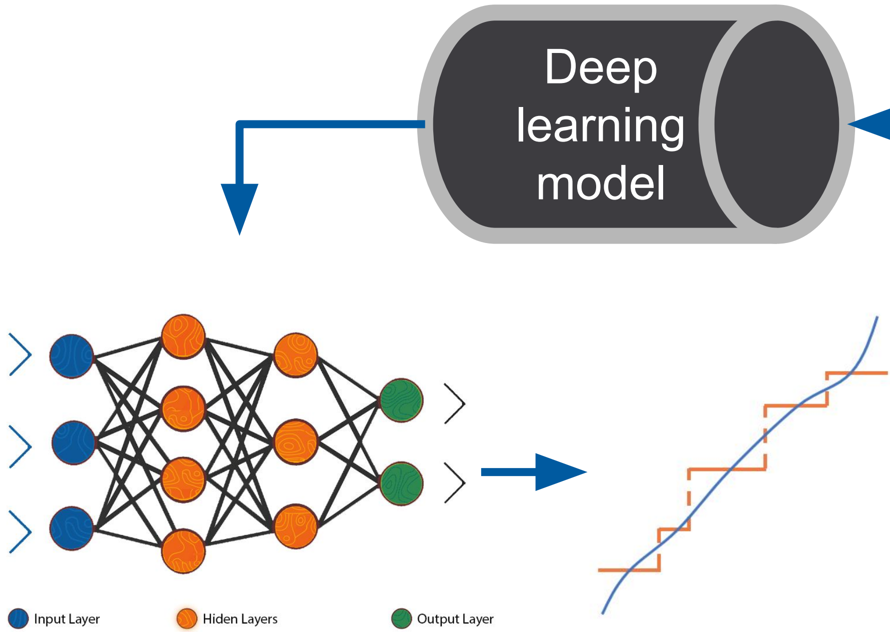
# Motivation

## Function approximation



# Motivation

## Function approximation




*“A 2-layer perceptron can approximate a continuous function to any desired accuracy…”*
G. Cybenko, 1989

model

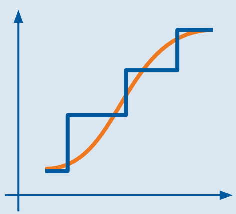
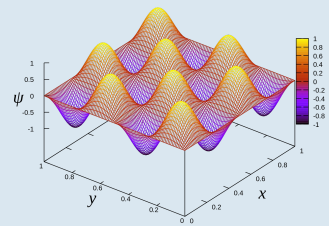
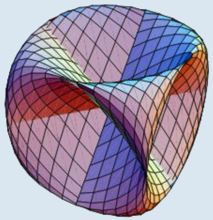
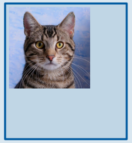
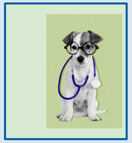
approximation

source: multilayer perceptron Archives - Analytics Vidhya
Cybenko, G.V. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2, 303-314.

# Motivation

## The curse of dimensionality

The amount of data $N$ needed to achieve the desired accuracy grows exponentially with the dimensionality $d$.

| | **1D example** | **2D example** | **3D example** | | **Image classification** |
|---|---|---|---|---|---|
| |  |   |  | … |   |
| # dim. | 1 | 2 | 3 | | ~ $10^3$ - $10^6$ |
| # data points | O(10) | O($10^2$) | O($10^3$) | | > #atoms in the universe |

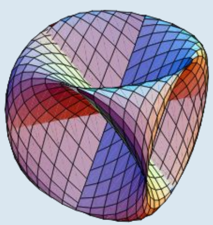
Source: https://mathworld.wolfram.com/SineSurface.html
https://commons.wikimedia.org/

# Motivation

## The curse of dimensionality

Universal approximation alone is not sufficient for scientific applications

**3D example**



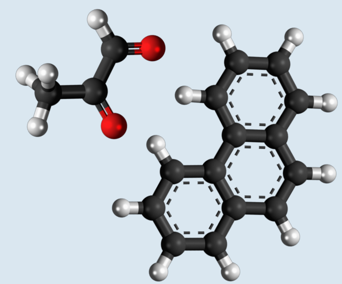
$d = 3$

…

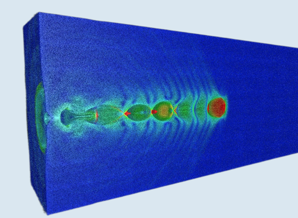
**Image classification**

$d \sim 10^3 - 10^6$

**Molecular dynamics**



**Physical processes**



$N = O(10^3)$

N > #atoms in the universe

out of question 😓

don't even ask 😥



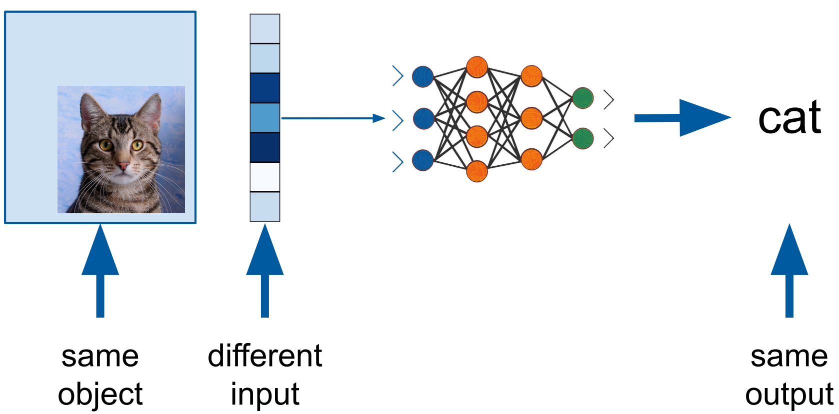
DRESDEN concept SCIENCE AND INNOVATION CAMPUS HZDR

# Motivation

## Symmetry in data

**Alternative**: incorporate symmetry to pre-defined transformations into a model

**Image classificati**




shift invariance must be learned from data

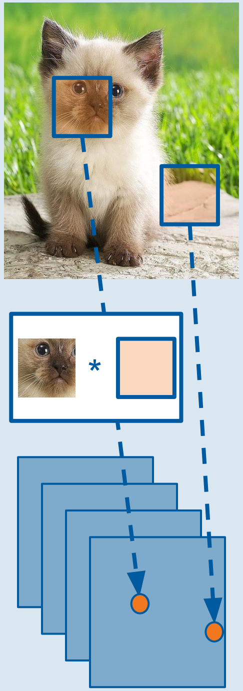
**augmentation**

- computationally inefficient;
- occupies network capacity;
- no theoretical guarantees.

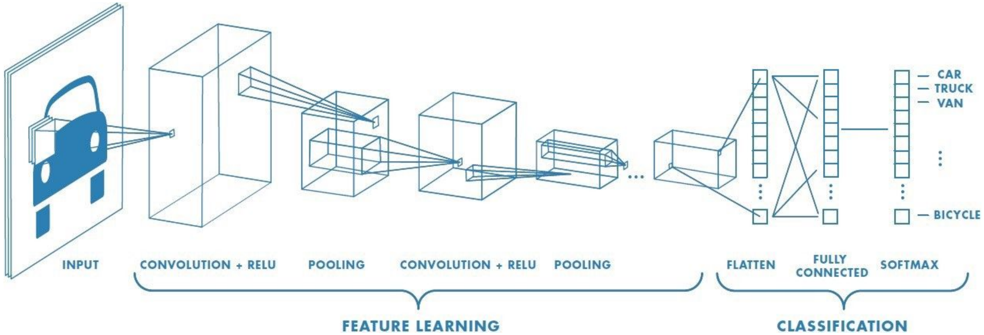
# Symmetry in deep learning

## Standard CNNs

### Convolution



### Convolutional neural networks





- LeCun, 1989
- translation equivariance
- weight sharing

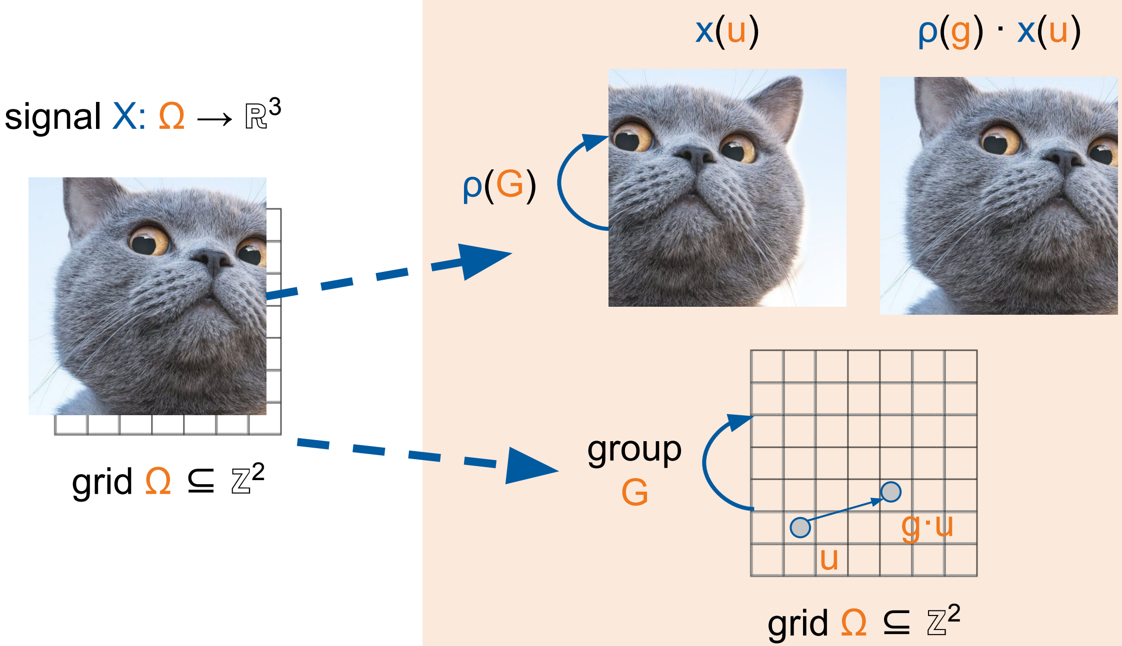
Source: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

# Symmetry in deep learning

## Symmetric prior

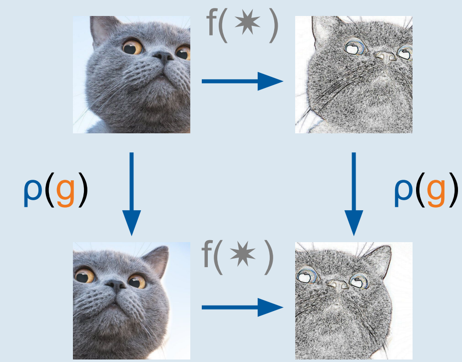
**Data**

**Symmetry transformation**




**Equivariance constraint**

$$\boxed{f(\rho(g)x) = \rho(g)\, f(x)}$$



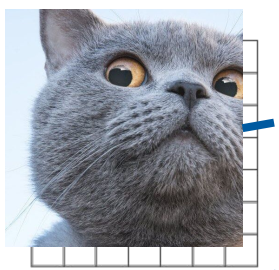

neural architecture must satisfy a G-equivariance constraint!

# Symmetry in deep learning

## Symmetric prior - translation

**Data**

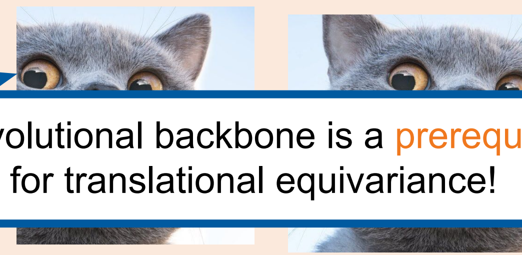
signal $X: \Omega \rightarrow \mathbb{R}^3$



grid $\Omega \subseteq \mathbb{Z}^2$

**Symmetry transformation**

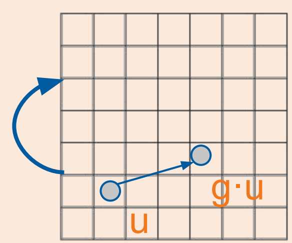
$x(u)$ $\rho(g) \cdot x(u)$



ρ

Convolutional backbone is a prerequisite for translational equivariance!

group $G$

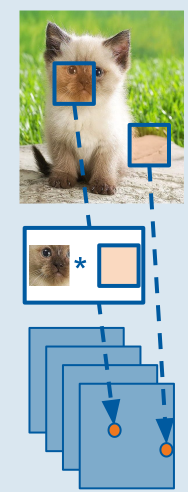



grid $\Omega \subseteq \mathbb{Z}^2$

**Neural architecture**

Conv. neural networks

$$(k * f)(x) = \sum_{y \in \mathbb{Z}^2} k(x-y) f(y)$$

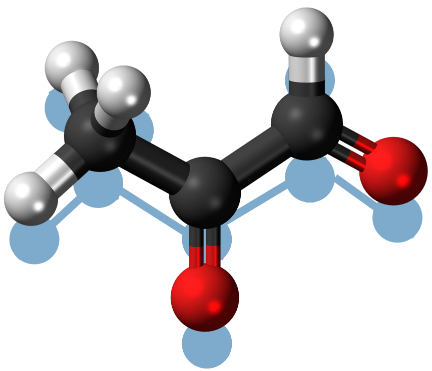


# Symmetry in deep learning

## Symmetric prior - translation + **permutation**

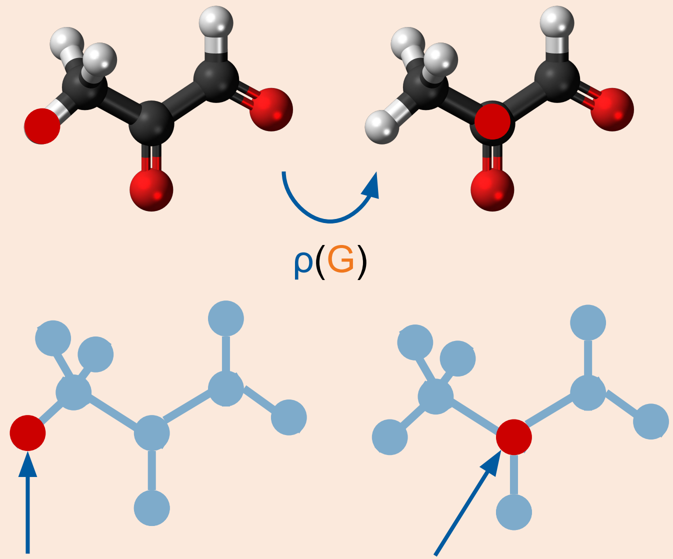
### Data

signal X: Ω → $\mathbb{R}^3$



graph Ω = (V,E)

### Symmetry transformation

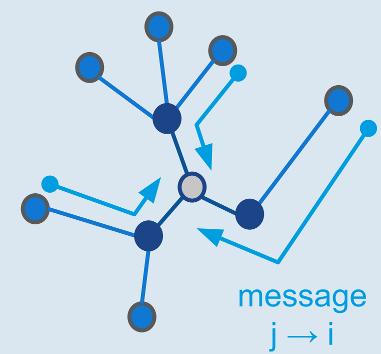



1st node    1st node

permutation group G

### Neural architecture
Graph neural networks

$$\boxed{(k * f)(x_i) = \sum_{j \in \mathcal{N}(i)} k(x_i - x_j) f(x_j)}$$




Source: https://commons.wikimedia.org/wiki/File:Methylglyoxal_molecule_ball.png

DRESDEN concept SCIENCE AND INNOVATION CAMPUS

HZDR

# Symmetry in deep learning

## Symmetric prior - translation + **rotation SO(n)**

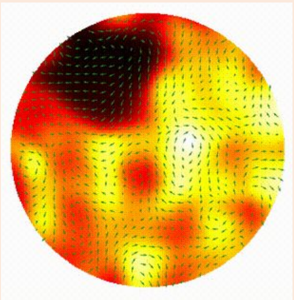
### Data

signal $X: \Omega \to \mathbb{R}^3$

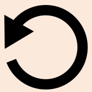
grid $\Omega \subseteq \mathbb{Z}^2$

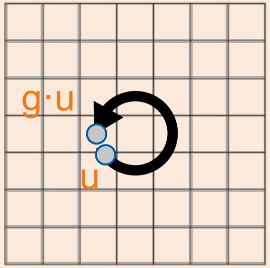
### Symmetry transformation

$\rho(g) \cdot x(u)$



$\rho(G)$






grid $\Omega \subseteq \mathbb{Z}^2$

### Neural architecture
Steerable CNNs (G-CNNs)

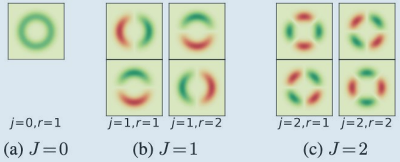
$$\boxed{\begin{aligned}(k * f)(x) &= \sum_{y \in \mathbb{Z}^2} k(x-y) f(y) \\ k(g.x) &= \rho_{out}(g) k(x) \rho_{in}(g)^T\end{aligned}}$$

### Solution

$$\phi(||x||) \cdot Y^J\left(\frac{x}{||x||}\right)$$

radial function

spherical harmonics

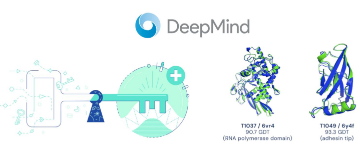



Source: https://github.com/QUVA-Lab/escnn
Weiler et al., 2018, Cesa et al., 2021

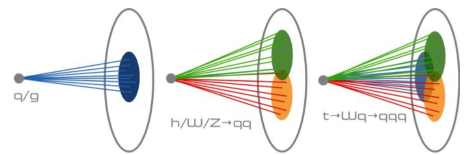
# Symmetry in deep learning

## Applications

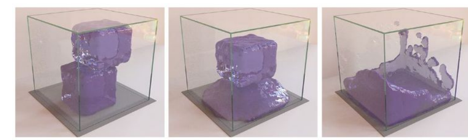
| | **Chemistry & Drug design** | **Physics** | **Simulations** |
|---|---|---|---|
| GNN | predicting protein's structure | particle jet classification task | physical simulations |
| G-CNN | molecular design | cryogenic EM | molecular dynamics |




Jumper et al., 2021

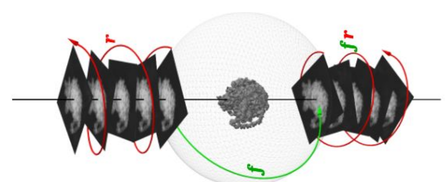



Murnane et al., 2022

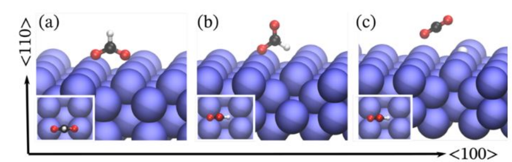


Sanchez-Gonzalez et al., 2020

Satorras et al., 2021



Cesa et al., 2022




Batzner et al., 2021

# Our contribution

## Symmetric prior - translation + **arbitrary group G**

**Neural architecture**

Steerable CNNs (G-CNNs)

$$(k * f)(x) = \sum_{y \in \mathbb{Z}^2} k(x-y) f(y)$$

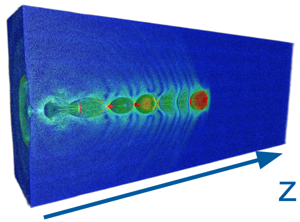
$$k(g.x) = \rho_{out}(g) k(x) \rho_{in}(g)^T$$

**Problem:**

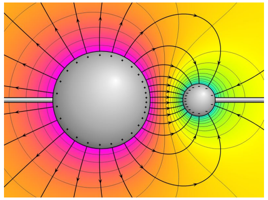
- so far, we introduced a solution for the group SO(n) of rotations only.
- In general the constraint on kernels must be solved *analytically* for each specific group G.

**Why should we care?**

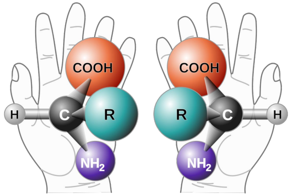
Many problems where O(n) symmetry breaks




propagation along z-axis



central-force problem




molecules

furniture

# Our contribution

## Symmetric prior - translation + **arbitrary group G**

**Neural architecture**
Steerable CNNs (G-CNNs)

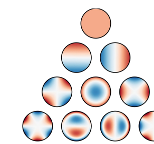
$$\boxed{\begin{aligned}(k * f)(x) &= \sum_{y \in \mathbb{Z}^2} k(x-y) f(y) \\ k(g.x) &= \rho_{out}(g) k(x) \rho_{in}(g)^T\end{aligned}}$$

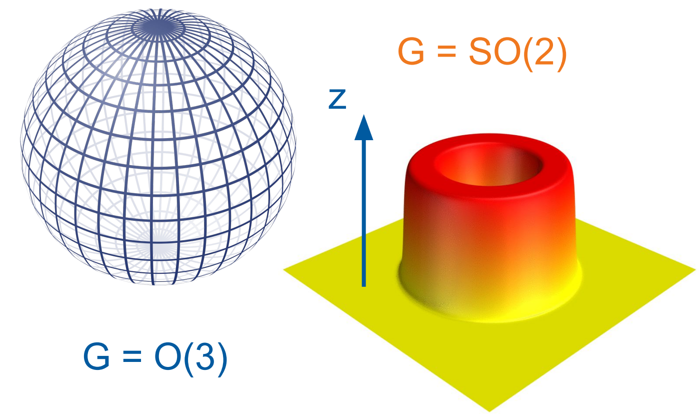
**Problem:**

- so far, we introduced a solution for the group SO(n) of rotations only.
- In general the constraint on kernels must be solved *analytically* for each specific group G.

**Solution (Cesa et al. 2021):**
adapt O(n)-steerable kernel bank to any sub-group by G-restriction.

**Warning:** the basis will be only suboptimal for a sub-group G.






Source: https://commons.wikimedia.org

# Our contribution

## Implicit representation of G-steerable kernels

**Neural architecture**

Steerable CNNs (G-CNNs)

$$\boxed{\begin{aligned}(k * f)(x) &= \sum_{y \in \mathbb{Z}^2} k(x-y) f(y) \\ k(g.x) &= \rho_{out}(g) k(x) \rho_{in}(g)^T\end{aligned}}$$

**Solution**

**Implicit Neural Convolutional Kernels for Steerable CNNs**


Maksim Zhdanov
Helmholtz-Zentrum Dresden-Rossendorf
maxxxzdn@gmail.com

Nico Hoffmann
Helmholtz-Zentrum Dresden-Rossendorf
n.hoffmann@hzdr.de

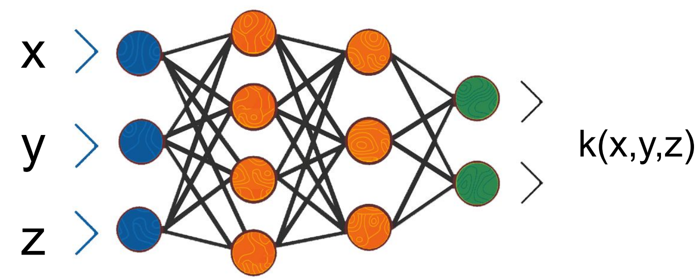
Gabriele Cesa
Qualcomm AI Research
University of Amsterdam
cesa.gabriele@gmail.com


use G-equivariant MLP to parameterize kernels
→ effortlessly develop G-custom kernel banks.

**G-equivariance of an MLP is all you need!**

**Lemma 1.** *If a kernel $k$ is parameterized by a G-equivariant MLP $\phi$ with input representation $\rho_{st}$ and output representation $\rho_{\otimes} := \rho_{in} \otimes \rho_{out}$, i.e. $vec(k)(x) := \phi(x)$, then the kernel satisfies the equivariance constraint in Equation 4 for a compact group G.*

$$\boxed{\text{MLP}(\rho(g)x) = \rho(g)\ \text{MLP}(x)}$$

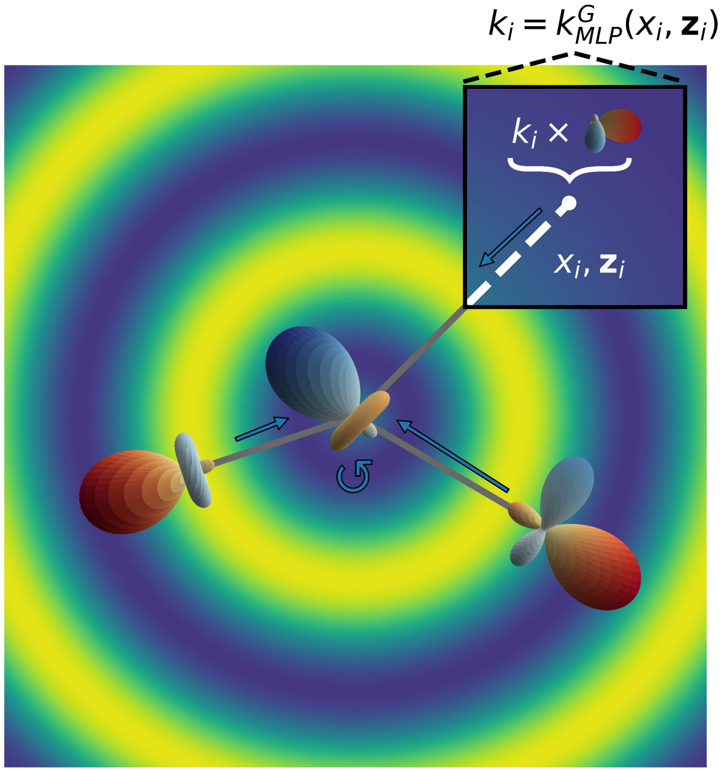



**Result**

learned kernel bank is optimal for G and a problem and easy to implement.

# Our contribution

## Additional kernel input




- a system might be endowed with *additional* relevant attributes, e.g.
  - physical quantities: atom type, bond type, mass, etc.
  - geometric quantities: distance, location, angle, etc.
- analytical solution for a kernel with additional information is *infeasible*.
- implicit representation allows one to introduce those features at *no cost*.

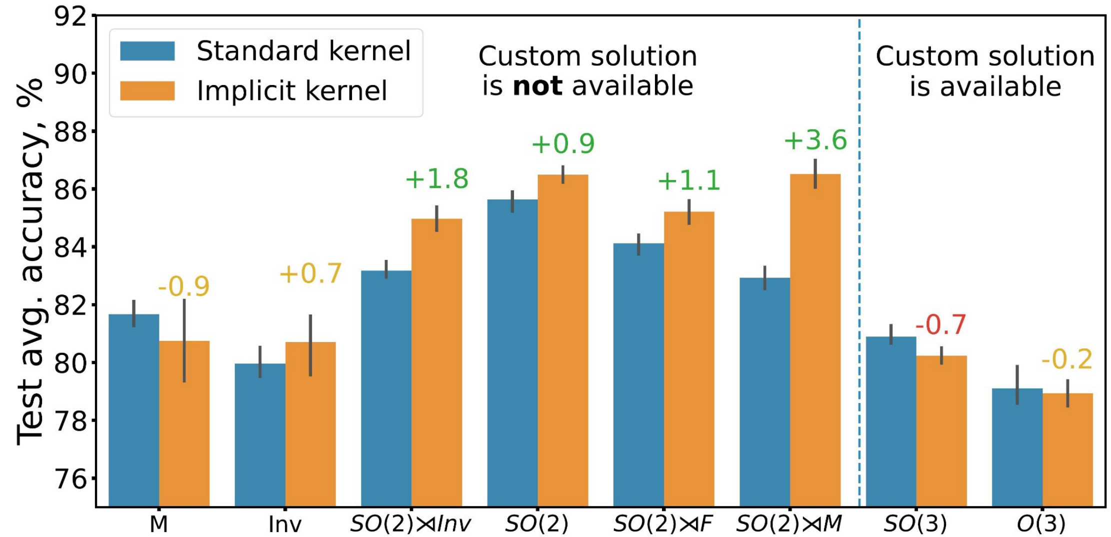
# Our contribution
## Results: point cloud data

**Data**: ModelNet-40 - point cloud models of furniture.

**Task**: predict the class of an object (40 classes in total).

**Goal:** show the performance gain of implicit kernels.

**Result:** IKs significantly *outperform* the baseline approach for most groups.

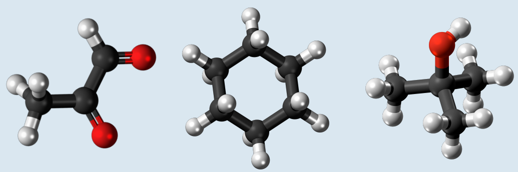


# Our contribution

## Results: molecular graph data

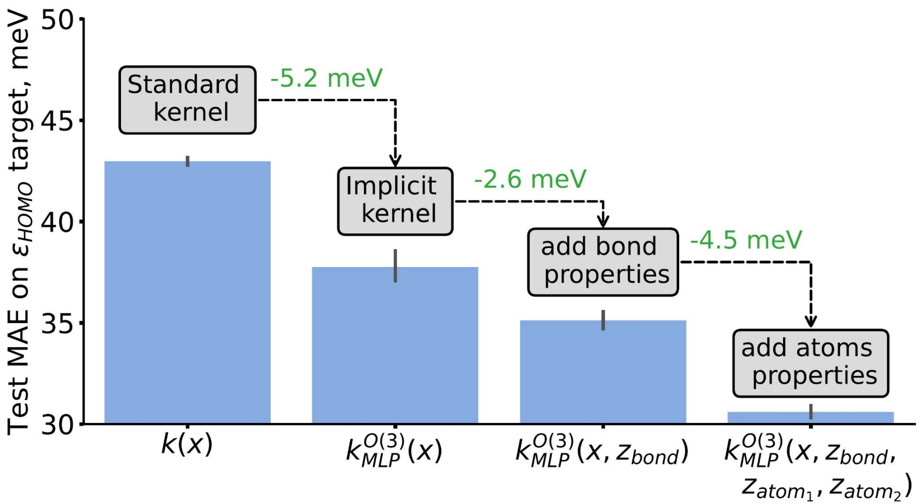
**Data**: QM9 - molecules represented as graphs.

**Task**: predict the quantum property of a molecule.



**Goal:** demonstrate the flexibility of implicit kernels.

**Result:** including conditional variables *drastically* improves performance of Steerable CNNs.

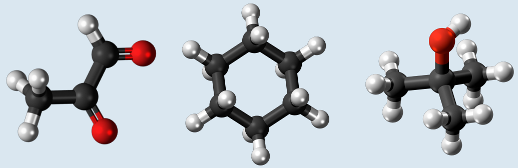


# Our contribution

## Results: molecular graph data

**Data**: QM9 - molecules represented as graphs.

**Task**: predict the quantum property of a molecule.



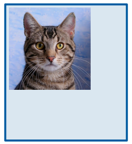
**Goal:** compare the performance of Steerable CNNs with implicit kernels to baselines.

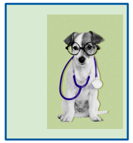
**Result:** IKs outperform linear steerable convolutions* but fall behind task-tailored models.

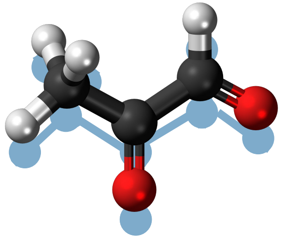
| Task | $\alpha$ | $\Delta\varepsilon$ | $\varepsilon_{HOMO}$ | $\varepsilon_{LUMO}$ | $\mu$ | $C_\nu$ | G | H | $R^2$ | U | $U_0$ | ZPVE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Units | bohr$^3$ | meV | meV | meV | D | cal/mol K | meV | meV | bohr$^3$ | meV | meV | meV |
| NMP [16] | .092 | 69 | 43 | 38 | .030 | .040 | 19 | 17 | 0.180 | 20 | 20 | 1.50 |
| SchNet [32] | .235 | 63 | 41 | 34 | .033 | .033 | 14 | 14 | 0.073 | 19 | 14 | 1.70 |
| Cormorant [1]* | .085 | 61 | 34 | 38 | .038 | .026 | 20 | 21 | 0.961 | 21 | 22 | 2.02 |
| L1Net [25]* | .088 | 68 | 46 | 35 | .043 | .031 | 14 | 14 | 0.354 | 14 | 13 | 1.56 |
| LieConv [12] | .084 | 49 | 30 | 25 | .032 | .038 | 22 | 24 | 0.800 | 19 | 19 | 2.28 |
| TFN [37]* | .223 | 58 | 40 | 38 | .064 | .101 | - | - | - | - | - | - |
| SE(3)-Tr. [15] | .142 | 53 | 35 | 33 | .051 | .054 | - | - | - | - | - | - |
| DimeNet++ [19] | .043 | 32 | 24 | 19 | .029 | .023 | 7 | 6 | 0.331 | 6 | 6 | 1.21 |
| SphereNet [23] | .046 | 32 | 23 | 18 | .026 | .021 | 8 | 6 | 0.292 | 7 | 6 | 1.12 |
| PaiNN [33] | .045 | 45 | 27 | 20 | .012 | .024 | 7 | 6 | 0.066 | 5 | 5 | 1.28 |
| EGNN [31] | .071 | 48 | 29 | 25 | .029 | .031 | 12 | 12 | 0.106 | 12 | 12 | 1.55 |
| SEGNN [5] | .060 | 42 | 24 | 21 | .023 | .031 | 15 | 16 | 0.660 | 13 | 15 | 1.62 |
| Ours | .078 | 45 | 24 | 22 | .033 | .032 | 21 | 19 | 0.809 | 19 | 19 | 2.08 |

# Conclusion

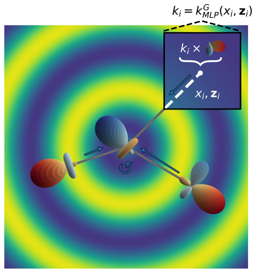
Symmetry is a *fundamental* design choice in modern deep learning

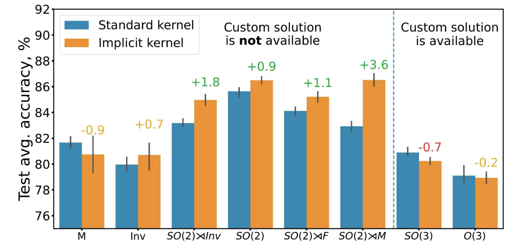






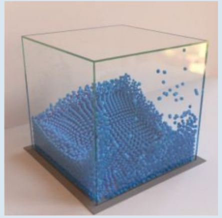
We propose a method for flexible and simple design of group equivariant CNNs



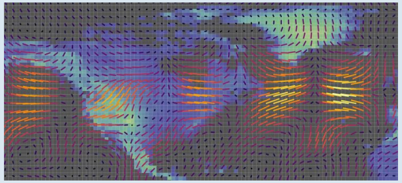

**There is a huge potential for applying symmetry-aware models. Reach out if you have something in mind!**

## physical simulations



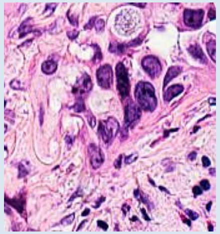
Battaglia et al., 2020

## climate science



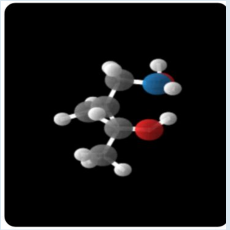
Brandstetter et al., 2022

## medical imaging



Bekkers et al., 2021

## density estimation



Satorras et al., 2021

## and many more…

- high energy physics;
- astronomy;
- PDE learning;
- quantum physics;
- gauge symmetry;
- …

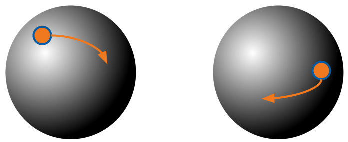
# Thank you for your attention!

Please do not hesitate to ask any questions :)

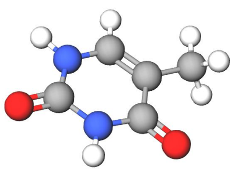
# Appendix

## Invariance & Equivariance

Let us have a symmetry group G

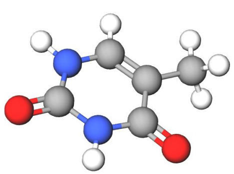


Let us consider a map $\Phi : X \rightarrow Y$

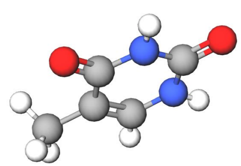


$\xrightarrow{\Phi}$ 852.20 kJ/mol

Let $T_g$ be a group action

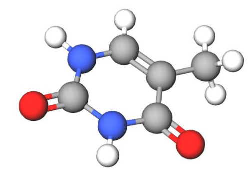


$\xrightarrow{T_g}$

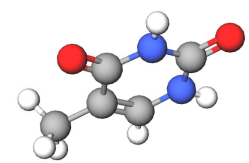


**Invariance**

Operator $\Phi$ is G-invariant if $\Phi \circ T_g = \Phi$



$\xrightarrow{\Phi}$ 852.20 kJ/mol

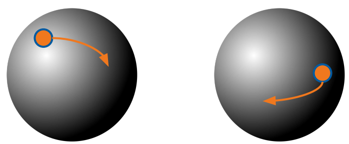


$\xrightarrow{\Phi \circ T_g}$ 852.20 kJ/mol

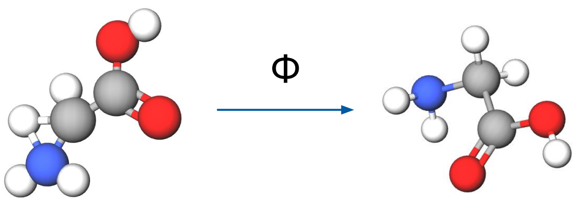
# Appendix

## Invariance & Equivariance

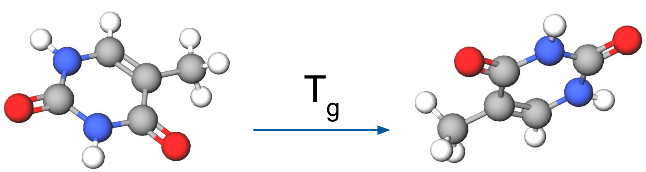
Let us have a symmetry group G



Let us consider a map $\Phi : X \to Y$

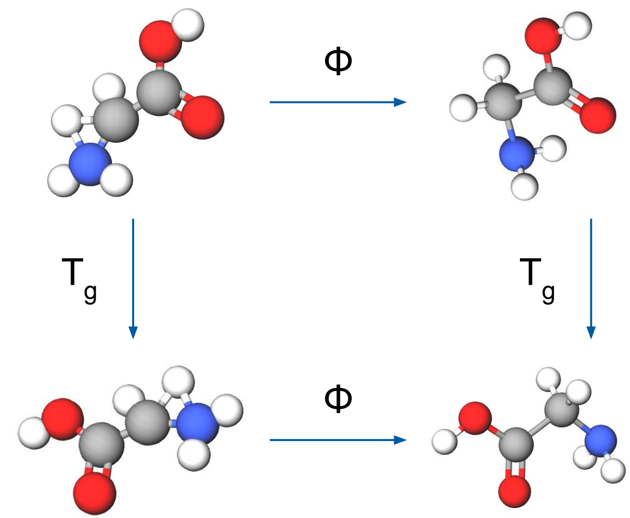



Let $T_g$ be a group action




**Equivariance**

Operator $\Phi$ is G-equivariant if $\Phi \circ T_g = T_g \circ \Phi$



# Appendix

## Implicit representation of G-steerable kernels

**Neural architecture**

Steerable CNNs (G-CNNs)

$$
\boxed{\begin{aligned}(k * f)(x) &= \sum_{y \in \mathbb{Z}^2} k(x-y)f(y) \\ k(g.x) &= \rho_{out}(g)k(x)\rho_{in}(g)^T\end{aligned}}
$$

**Solution**

**Implicit Neural Convolutional Kernels for Steerable CNNs**


Maksim Zhdanov
Helmholtz-Zentrum Dresden-Rossendorf
maxxxzdn@gmail.com

Nico Hoffmann
Helmholtz-Zentrum Dresden-Rossendorf
n.hoffmann@hzdr.de

Gabriele Cesa
Qualcomm AI Research
University of Amsterdam
cesa.gabriele@gmail.com


use G-equivariant MLP to parameterize kernels
→ effortlessly develop G-custom kernel banks.

**Schematic overview**

Implicit G-steerable kernel