ILPO-NET: Network for the invariant recognition of arbitrary volumetric patterns in 3D

Dmitrii Zhemchuzhnikov, Sergei Grudinin

ILPO-NET: Network for the invariant recognition of arbitrary volumetric patterns in 3D

Dmitrii Zhemchuzhnikov, Sergei Grudinin

Univ. Grenoble Alpes, CNRS, Grenoble INP, LJK, Grenoble, France
{dmitrii.zhemchuzhnikov,sergei.grudinin}@univ-grenoble-alpes.fr

Abstract. Effective recognition of spatial patterns and learning their hierarchy is crucial in modern spatial data analysis. Volumetric data applications seek techniques ensuring invariance not only to shifts but also to pattern rotations. While traditional methods can readily achieve translational invariance, rotational invariance possesses multiple challenges and remains an active area of research. Here, we present ILPO-Net (Invariant to Local Patterns Orientation Network), a novel approach that handles arbitrarily shaped patterns with the convolutional operation inherently invariant to local spatial pattern orientations using the Wigner matrix expansions. Our architecture seamlessly integrates the new convolution operator and, when benchmarked on diverse volumetric datasets such as MedMNIST and CATH, demonstrates superior performance over the baselines with significantly reduced parameter counts – up to 1000 times fewer in the case of MedMNIST. Beyond these demonstrations, ILPO-Net's rotational invariance paves the way for other applications across multiple disciplines. Our code is publicly available at https://gricad-gitlab.univ-grenoble-alpes.fr/GruLab/ILPO/-/tree/main/ILPONet.

Keywords: Volumetric data · 3DCNN · Pattern recognition · Rotational invariance · SO(3) invariance · SE(3) invariance.

1 Introduction

In the constantly evolving world of data science, three-dimensional (3D) data models have emerged as a focal point of academic and industrial research. As the dimensionality of data extends beyond traditional 1D signals and 2D images, capturing the third dimension opens new scientific challenges and brings various opportunities. The possible applications of new methods range from sophisticated 3D models in computer graphics to the analysis of volumetric medical scans.

With the advent of deep learning, techniques that once revolutionized two-dimensional image processing are now being adapted and extended to deal with the volumetric nature of 3D data. However, the addition of the third dimension not only increases the computational complexity but also opens new theoretical challenges. One of the most pressing ones is the need for persistent treatment of volumetric data in arbitrary orientation. A particular example is medical imaging, where the alignment of a scan may vary depending on the equipment, the technician, or even the patient.

However, achieving such rotational consistency is non-trivial. While data augmentation techniques, such as artificially rotating training samples, can help to some extent, they do not inherently equip a neural network with the capability to recognize rotated patterns. Moreover, such methods can significantly increase the computational cost, both at the training and inference time, especially with high-resolution 3D data. The community witnessed a spectrum of novel approaches specifically designed for these challenges. As we will see below, they range from modifications of traditional convolutional networks to the introduction of entirely new paradigms built on advanced mathematical principles.

This paper presents a novel approach to *invariant* pattern recognition in *regular volumetric data*. In contrast to other methods, our convolution operation maps from 3D to 3D space without constraints on the filter shape. We shall note that pattern recognition can be invariant or equivariant to the pattern orientation. The equivariant approach generally allows for a better expressivity of the model but requires more model parameters and additional dimensions in the output map to memorize pattern orientations. The invariant approach may lack expressiveness but enables staying in the 3D space with much fewer model parameters.

2 Related Work

Neural networks designed to process spatial data learn the data hierarchy by detecting local patterns and their relative position and orientation. However, when dealing with data in two or more dimensions, these patterns can be oriented arbitrarily, which makes neural network predictions dependent on the orientation of the input data. Classically, this dependence can be bypassed through data augmentation with rotated copies of data samples in the training set [18]. In the volumetric (3D) case, augmentation often results in significant extra computational costs. For some types of three-dimensional data, the canonical orientation of data samples or their local volumetric patterns can be uniquely defined [22,11,9,36]. In most real-world scenarios, though, it is common for 3D data to be oriented arbitrarily. Thus, there was a pressing need for methods with specific properties of rotational invariance or equivariance by design. We can trace two main directions in the development of these methods: those based on equivariant operations in the SO(3) space (space of rotations in 3D), and those with learnable filters that are orthogonal to the SO(3) or SE(3) (roto-translational) groups.

The pioneering method from the first class was the Group Equivariant Convolutional Networks (G-CNNs) introduced by Cohen et al. [4] , who proposed a general view on convolutions in different group spaces. Many more methods were built up subsequently upon this approach [32,31,2,29,23,6,24,14,21]. Several implementations of Group Equivariant Networks were specifically adapted for regular volumetric data, e.g., CubeNet[32] and 3D G-CNN [31]. The authors of these methods consider a discrete set of 90-degree rotations and reflections, which exhaustively describe the possible positions of a cubic pattern on a regular grid.

However, we shall note that, typically, both discrete and regular data are representations of the continuous realm, which embodies a continuous range of rotations. As a result, they cannot be reduced to just a finite series of 90-degree turns. Another limitation is that this group of methods performs summing over rotations that can lead to the higher output of radially-symmetrical filters, which limits the expressiveness of the models because the angular dependencies of patterns are not memorized in the filters , as we show in Appendix B. Another branch in this development direction was represented by methods aimed at detecting patterns on a sphere. In Spherical CNNs, Cohen et al. proposed a convolution operation defined on the spherical surface, making it inherently rotationally equivariant [5]. Spherical CNNs are a comprehensive tool for working with spherical data, but they have limited application to volumetric cases. When thinking of expanding this approach for volumetric data where each voxel possesses its own coordinate system, there remains the challenge of information exchange between different spheres.

Let us characterize methods from the second class without delving deeply into mathematical terms. Here, each layer of the network operates with products of pairs of oriented input quantities. These products inherit the orientation of the input, and then they are summed up with learnable weights. The first two methods to mention in this section are the Tensor Field Networks (TFN) [28] and the N-Body Networks (NBNs) [16]. Kondor [17] presented a similar approach, the Clebsch-Gordan Nets applied to data on a sphere. These models employed spherical tensor algebra working on irregular point clouds. Weiler et al. [30] proposed 3D Steerable CNNs, where they applied the same algebra to regular voxelized data. These three methods impose constraints on the trainable filters and consider only equivariant filter subgroups. As a result, they may not discriminate some patterns. , as we show in Appendix A A. Satorras et al. [27] built EGNN on the same idea. However, they achieved equivariance in a much simpler but less expressive way without the usage of Clebsch-Gordan coefficients and spherical harmonics. It is also worth mentioning the works of [25,19,35], where the authors applied the same logic to geometric algebra but used multivectors instead of irreducible representations of S^2.

Apart from the two main directions, we can highlight the application of differential geometry, such as moving frames, to volumetric data, as demonstrated by Sangalli et al. [26]. This approach uses local geometry to set up the local pattern orientation. This idea unites the method with the family of Gauge networks [3]. The current implementation still depends on the discretization of input data. Rotating input samples can significantly reduce accuracy, as shown in [26].

Considering the points mentioned above, there is a need to create a technique that can detect local patterns of any shape in input 3D data, regardless of their orientation. This method should approach spaces $\mathbb{R}^3$ and SO(3) differently. While operating in $\mathbb{R}^3$ requires a convolution, one shall avoid summation over orientations in the rotational space. Andrearczyk et al. followed this approach in [1], but they restricted the shape of the learnable filters. Additionally, their

method has a narrow application domain, whereas we intend to develop a data-generic technique. Below, we propose a novel convolutional operation, Invariant to Local Features Orientation Network layer, that can detect arbitrary volumetric patterns, regardless of their orientations. This operation can be used in any convolutional architecture without substantial modifications. Our experiments on several datasets, CATH and the MedMNIST collection, demonstrate that this operation can achieve higher accuracy than the state-of-the-art methods with up to 3 orders of magnitude fewer learnable parameters. To summarize, our contributions are:

1. In contrast to previous approaches, our method detects arbitrary-shaped filters in regular volumetric data;
2. We propose a rotational pooling operation that considers continuous space of rotations and avoids summing up in the rotational space;
3. The novel convolution can be used in any convolutional architecture without other modifications.

3 Theory

3.1 Problem statement

The conventional 3D convolution can be formally expressed as:

$$h(\boldsymbol{\Delta}) = \int_{\mathbb{R}^3} f(\boldsymbol{r} + \boldsymbol{\Delta}) g(\boldsymbol{r}) d\boldsymbol{r}, \tag{1}$$

where $f(\boldsymbol{r})$ is a function describing the input data, $g(\boldsymbol{r})$ is a filter function, and $h(\boldsymbol{\Delta})$ is the convolution output function that depends on the position of the filter with respect to the original data $\boldsymbol{\Delta}$. The meaning of this operation in light of pattern recognition is that the value of the overlap integral of the filter and the fragment of the input data map around point $\boldsymbol{\Delta}$ serves as an indicator of the presence of the pattern in this point. However, such a recognition works correctly only if the orientation of the pattern in the filter and in the input data coincide. Therefore, if the applied pattern has a wrong orientation, a conventional convolution operation cannot recognize it.

The logical solution would be to apply the filter in multiple orientations. Then, the orientation of the filter appears in the arguments of the output function. In this approach, we consider a convolution with a rotated filter, represented as $g(\mathcal{R}\boldsymbol{r})$, where $\mathcal{R} \in \mathrm{SO}(3)$,

$$h(\boldsymbol{\Delta}, \mathcal{R}) = \int_{\mathbb{R}^3} f(\boldsymbol{r} + \boldsymbol{\Delta}) g(\mathcal{R}\boldsymbol{r}) d\boldsymbol{r}. \tag{2}$$

The outcome of this convolution depends on both the shift $\boldsymbol{\Delta}$, and the filter rotation $\mathcal{R}$. The output function $h(\boldsymbol{r}, \mathcal{R})$ is now defined in $6D$ but if we want to obtain a 3D map that indicates that a pattern $g(\boldsymbol{r})$ in arbitrary orientation

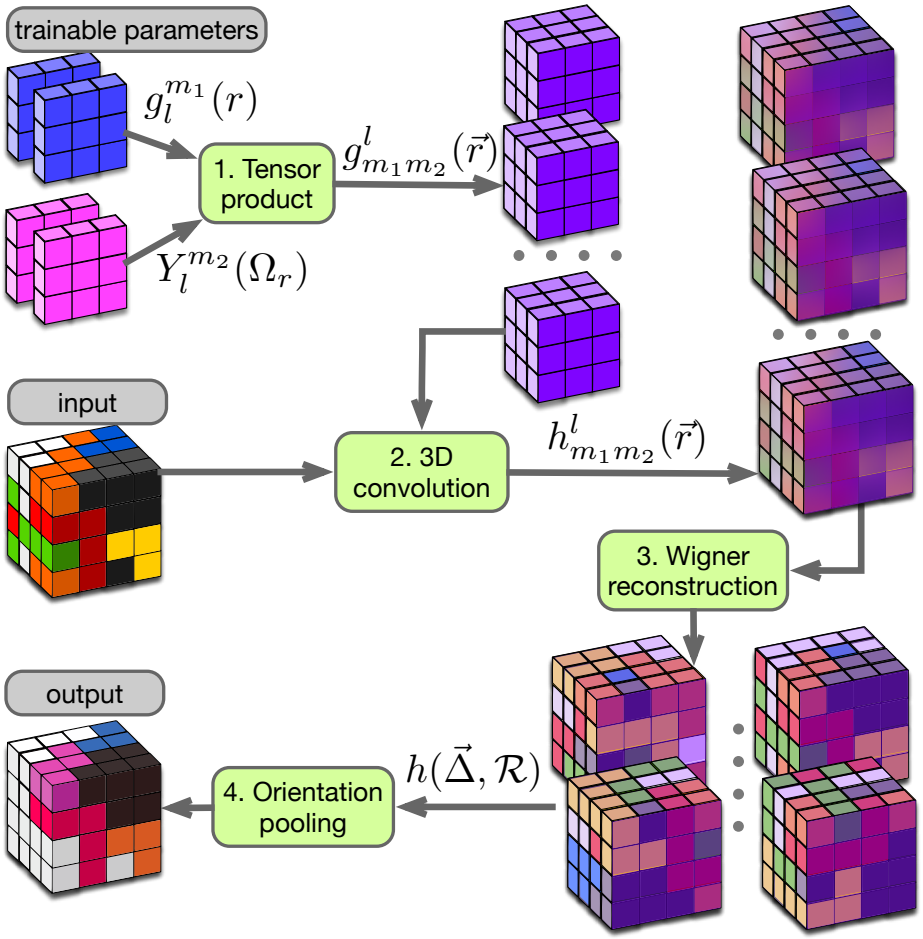

Fig. 1. Schematic illustration of the ILPO convolution. The diagram showcases the main steps involved in our convolution process: 1) Tensor product of trainable filter coefficients and spherical harmonics; 2) 3D convolution of the input image and the rotated filter coefficients; 3) Reconstruction of the convolution output in the SO(3) space; 4) Orientation (soft)-max pooling.

was detected at a point $\boldsymbol{\Delta}$ of map $f(\boldsymbol{r}+\boldsymbol{\Delta})$, we need to conduct an additional *orientation pooling* operation:

$$h(\boldsymbol{\Delta}) = \text{OrientionPool}_{\mathcal{R}}[h(\boldsymbol{\Delta}, \mathcal{R})], \tag{3}$$

which can generally be defined in different ways. The only constraint on this operation is that it must be *rotationally invariant* with respect to $\mathcal{R}$ or, in the discrete case, *invariant to the permutation* of the set of rotations:

$$\text{OrientionPool}_{\mathcal{R}}[f(\mathcal{R})] = \text{OrientionPool}_{\mathcal{R}}[f(\mathcal{R}\mathcal{R}')] \quad \forall f \quad \text{and} \quad \forall \mathcal{R}'. \tag{4}$$

The simplest pooling operation satisfying this constraint would be an average over orientations $\mathcal{R}$. However, this will be equivalent to averaging the filter $g(\boldsymbol{r})$ over all possible orientations. Such an averaged filter is radially symmetric and is thus not very expressive. A better $\text{OrientionPool}_{\mathcal{R}}$ operation would be extracting a maximum over orientations $\mathcal{R}$ or applying a softmax operation, as defined below,

$$\begin{aligned} \max_{\mathcal{R}} f(\mathcal{R}) &= \lim_{n\to\infty} \sqrt[n]{\int_{\text{SO(3)}} f^n(\mathcal{R}) d\mathcal{R}} \\ \text{softmax}_{\mathcal{R}} f(\mathcal{R}) &= \frac{\int_{\text{SO(3)}} \text{relu}(f(\mathcal{R}))^2 d\mathcal{R}}{\int_{\text{SO(3)}} \text{relu}(f(\mathcal{R})) d\mathcal{R}} \end{aligned} . \tag{5}$$

Attempting to incorporate such a convolution in a neural network, we face several challenges.

1. If we assume $g(\boldsymbol{r})$ to be a learnable filter, it is not trivial to guarantee the correct backpropagation from multiple orientations of the filter to the original orientation of the filter $g(\boldsymbol{r})$.
2. Finding the hard- or soft- maximum in the pooling operation in the discrete case requires a consideration of a large number of rotations in the SO(3) space to reduce the deviation of the sampling maximum from the true maximum. To make the method feasible we need to avoid performing the 3D convolution for each of these rotations.

3.2 Method

Any square-integrable function on a unit sphere $g(\Omega) : S^2 \to \mathbb{R}$ can be expanded as a linear combination of spherical harmonics $Y_l^m(\Omega)$ of degrees l and orders k as

$$g(\Omega) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} g_l^m Y_l^m(\Omega). \tag{6}$$

The expansion coefficients f_l^m can then be obtained by the following integrals,

$$g_l^m = \int_{S^2} g(\Omega) Y_l^m(\Omega) d\Omega. \tag{7}$$

Wigner matrices $D^l_{m_1m_2}(\mathcal{R})$ are defined for $\mathcal{R} \in \mathrm{SO}(3)$ and provide a representation of the rotation group SO(3) in the space of spherical harmonics:

$$Y_l^{m_1}(\mathcal{R}\Omega) = \sum_{m_2=-l}^{l} D^l_{m_1m_2}(\mathcal{R}) Y_l^{m_2}(\Omega). \tag{8}$$

Since Wigner matrices are orthogonal, i.e.,

$$\int_{\mathrm{SO}(3)} D^l_{m_1m_2}(\mathcal{R}) D^{l'}_{k_1'k_2'}(\mathcal{R}) d\mathcal{R} = \frac{8\pi^2}{2l+1} \delta_{ll'} \delta_{m_1m_1'} \delta_{m_2m_2'}, \tag{9}$$

any square-integrable function $h(\mathcal{R}) \in L^2(\mathrm{SO}(3))$ can be decomposed into them as

$$h(\mathcal{R}) = \sum_{l=0}^{\infty} \sum_{m_1=-l}^{l} \sum_{m_2=-l}^{l} h^l_{m_1m_2} D^l_{m_1m_2}(\mathcal{R}), \tag{10}$$

where the expansion coefficients $h^l_{m_1m_2}$ are obtained by integration as

$$h^l_{m_1m_2} = \frac{2l+1}{8\pi^2} \int_{\mathrm{SO}(3)} h(\mathcal{R}) D^l_{m_1m_2}(\mathcal{R}) d\mathcal{R}. \tag{11}$$

Let us now consider the following decomposition of a function $h(\boldsymbol{\Delta}, \mathcal{R})$,

$$h(\boldsymbol{\Delta}, \mathcal{R}) = \sum_{l,m_1,m_2} h^l_{m_1m_2}(\boldsymbol{\Delta}) D^l_{m_1m_2}(\mathcal{R}). \tag{12}$$

Inserting the spherical harmonics decomposition of the rotated kernel $g(\mathcal{R}\boldsymbol{r})$ in Eq. 2, we obtain

$$\begin{aligned} h(\boldsymbol{\Delta}, \mathcal{R}) = \int_{\mathbb{R}^3} f(\boldsymbol{r} + \boldsymbol{\Delta}) \sum_{lm_1} g_l^{m_1}(r) Y_l^{m_1}(\mathcal{R}\Omega_r) d\boldsymbol{r} = \\ \int_{\mathbb{R}^3} f(\boldsymbol{r} + \boldsymbol{\Delta}) \sum_{lm_1} g_l^{m_1}(r) \sum_{m_2} D^l_{m_1m_2}(\mathcal{R}) Y_l^{m_2}(\Omega_r) d\boldsymbol{r}, \end{aligned} \tag{13}$$

where (r, Ω_r) are the radial and the angular components of the vector $\boldsymbol{r}$. Changing the order of operations, we get the following expression,

$$h(\boldsymbol{\Delta}, \mathcal{R}) = \sum_{lm_1m_2} D^l_{m_1m_2}(\mathcal{R}) \int_{\mathbb{R}^3} f(\boldsymbol{r} + \boldsymbol{\Delta}) g^l_{m_1m_2}(\boldsymbol{r}) d\boldsymbol{r}, \tag{14}$$

where we introduce expansion coefficients $g^l_{m_1m_2}(\boldsymbol{r})$ at a point $\boldsymbol{r}$ with the radial and angular components (r, Ω_r) as

$$g^l_{m_1m_2}(\boldsymbol{r}) = g_l^{m_1}(r) Y_l^{m_2}(\Omega_r). \tag{15}$$

Consequently, equating Eq. 12 to Eq. 14 and applying orthogonal conditions from Eq. 9 on both sides, we obtain

$$h^l_{m_1m_2}(\boldsymbol{\Delta}) = \int_{\mathbb{R}^3} f(\boldsymbol{\Delta} + \boldsymbol{r}) g^l_{m_1m_2}(\boldsymbol{r}) d\boldsymbol{r}. \tag{16}$$

In summary, our method comprises four steps, as depicted in Figure 1:

1. **Tensor product** of $g_l^{m_1}(r)$ and $Y_l^{m_2}(\Omega_r)$, where $g_l^{m_1}(r)$ are learnable filters (see Eq. 15).
2. **3D convolution** involving $g^l_{m_1 m_2}(\boldsymbol{r})$ and $f(\boldsymbol{r})$, with $f(\boldsymbol{r})$ representing the input data (refer to Eq 16).
3. **Wigner reconstruction** of $h(\boldsymbol{\Delta}, \mathcal{R})$ (see Eq. 12).
4. **Orientation pooling** as detailed in Eq. 5.

By employing these steps, we reduce the computational complexity through the utilization of Wigner matrices following the 3D convolution. The connection between the number the sampled points in SO(3) and the number of coefficients is elaborated upon in Appendix C. Furthermore, subsection 4.1 presents an empirical examination of how these quantities influence the maximum sampling error. Subsection 3.3 provides details of the implementation of the method in the discrete case.

3.3 Implementation for the voxelized data

Discrete convolution Here we describe how the convolution introduced above can be discretized for use in a neural network with *regular voxelized data*. Let us firstly define for each filter $g(\boldsymbol{r})$, where $\boldsymbol{r} = (x_i, y_j, z_k)$, a regular Cartesian grid of a linear size L: $0 \leq i, j, k < L$. This size also defines the maximum expansion order of the spherical harmonics expansion in Eq. 6. Let us also compute spherical coordinates (r_{ijk}, Ω_{ijk}) for a voxel with indices i, j, k in the Cartesian grid with respect to the center of the filter. For each of data voxel of radii r_{ijk} inside the filter grid, with the origin in the center of the grid, we define a filter $g^l_{m_1 m_2}(x_i, y_j, z_k)$ and parameterize it with learnable coefficients $g_l^m(r_{ijk})$ and non-learnable spherical harmonics basis functions according to Eq. 15 .

$$g^l_{m_1 m_2}(x_i, y_j, z_k) = g_l^{m_1}(r_{ijk}) Y_l^{m_2}(\Omega_{ijk}). \tag{17}$$

After, we conduct a *discrete* version of the 3D convolution from Eq. 16:

$$h^l_{m_1 m_2}(x_i, y_j, z_k) = \sum_{i'=0}^{L-1} \sum_{j'=0}^{L-1} \sum_{z'=0}^{L-1} f(x_{i+i'-L//2}, y_{j+j'-L//2}, z_{k+k'-L//2}) \\ g^l_{m_1 m_2}(x_{i'}, y_{j'}, z_{k'}), \tag{18}$$

where $f(x_i, y_j, z_k)$ is the input voxelized data. This operation has a complexity of $O(N^3 \times D_{\text{in}} \times D_{\text{out}} \times L^6)$, where the multiplier L^6 is composed of the size of the filter, L^3, and the number of $g^l_{m_1 m_2}$ coefficients $\propto L^3$, N is the linear size of the input data $f(\boldsymbol{r})$ and D_{in} and D_{out} are the number of the input and the output channels, respectively. For the computational efficiency of our method, we always keep the value of L fixed and small, independent of N.

To perform the Wigner matrix reconstruction in Eq. 10, we need to numerically integrate the SO(3) space. We can compute this integral *exactly* using the Gauss-Legendre quadrature scheme from L points [12]. It is convenient to represent a rotation in SO(3) by a successive application of three Euler angles α, β and γ,

about the axes Z, Y and Z, respectively. Then, the Wigner matrix $D^l_{m_1m_2}(\mathcal{R})$ can be expressed as a function of three angles: $D^l_{m_1m_2}(\mathcal{R}) = D^l_{m_1m_2}(\alpha, \beta, \gamma)$ and written as a sum of two terms:

$$\begin{aligned} D^l_{m_1m_2}(\alpha, \beta, \gamma) = C_{m_1}(m_1\alpha)[d_1]^l_{m_1m_2}(\beta)C_{m_2}(m_2\gamma) \\ +C_{-m_1}(m_1\alpha)[d_2]^l_{m_1m_2}(\beta)C_{-m_2}(m_2\gamma), \end{aligned} \tag{19}$$

where $[d_i]^l_{m_1m_2}, i = 1, 2$ can be decomposed into associated Legendre polynomials $P^m_l(\cos(\beta)), 0 \leq m < l$, and C_m is defined as follows:

$$C_m(x) = \begin{cases} \cos(x), & m \geq 0 \\ \sin(x), & m < 0 \end{cases}. \tag{20}$$

Given such a form of $D^l_{m_1m_2}(\alpha, \beta, \gamma)$, we discretize the space of rotations SO(3) as a 3D space with dimensions along the α, β and γ angles. The dimensions α and γ have a regular division. We use the Gauss–Legendre quadrature to discretize $\cos(\beta)$ to define the β dimension. Then, we perform the discrete version of the summation in Eq. 12:

$$\begin{aligned} h(x_i, y_j, z_k, \alpha_q, \beta_r, \gamma_s) = \sum_{m_2=-l}^{l} C_{m_2}(m_2\gamma_s)(\sum_{m_1=-l}^{l} C_{m_1}(m_1\alpha_q) \\ (\sum_{l=0}^{L-1}[d_1]^l_{m_1m_2}(\beta_r)h^l_{m_1m_2}(x_i, y_j, z_k))) + \sum_{m_2=-l}^{l} C_{-m_2}(m_2\gamma_s) \\ (\sum_{m_1=-l}^{l} C_{-m_1}(m_1\alpha_q)(\sum_{l=0}^{L-1}[d_2]^l_{m_1m_2}(\beta_r)h^l_{m_1m_2}(x_i, y_j, z_k))), \end{aligned} \tag{21}$$

where $0 \leq q, r, s \leq K - 1$, K is the linear size of the SO(3) space discretization. If we assume that $L < K$, then the complexity of the reconstruction is $O(N^3 \times D_{\text{out}} \times K^3 \times L)$, where N is the linear size of the input data $f(\boldsymbol{r})$, and D_{in} and D_{out} are the number of the input and the output channels, respectively. We shall specifically note that this operation has a lower complexity compared to the case of Eq. 2, if the latter is calculated with a brute-force approach provided that the number of sampled points in the SO(3) space $K^3 >> L^3$.

Orientation pooling For the orientation pooling operation, we have considered two nonlinear operations, hard maximum and soft maximum defined in Eq. 5. While only L^3 points in the SO(3) space are sufficient to find the exact value of the integration of functions $h(x_i, y_j, z_k, \alpha, \beta, \gamma)$, many more points are required to approximate the integration of $\text{relu}(h(x_i, y_j, z_k, \alpha, \beta, \gamma))^2$ or $h^n(x_i, y_j, z_k, \alpha, \beta, \gamma)$. There is not a closed-form dependency between K, L and ϵ, the error of discrete approximation of integrals in Eq. 5 on the grid of K^3 points. However, we need to ensure that the deviation of the sampling maximum from the real maximum for a given sampling division K is bounded. For this purpose we introduce lemmas and theorems in Appendix C.

4 Results

As mentioned in the introduction, we have specifically designed our model for regular volumetric data. Benchmarking our method on irregular representation would require significant modifications of the model that are out of the scope of the present paper. Therefore, we chose two representative benchmarks from different application domains: CATH and MedMNIST3D, described below in more detail. We also conducted additional experiments to examine the properties of our operations.

4.1 Orientation invariance

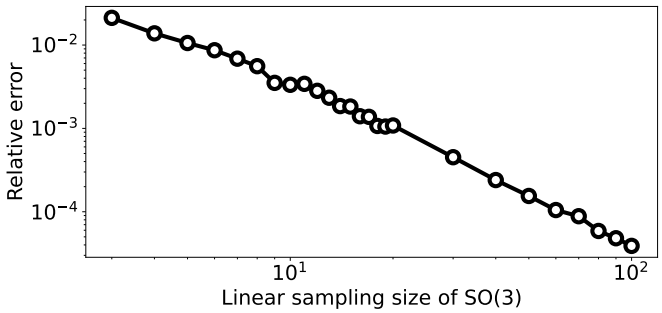

Fig. 2. Standard deviation of sampled maxima relative to the true function maximum (y-axis) as a function of sampling size K in the SO(3) space (x-axis).

To investigate the sensitivity of orientation-independent pattern detection to the linear size of sampling, we conducted the following experiment. We initiate a function in the SO(3) space with a Wigner matrix decomposition up to a maximum degree of 2 ($L = 3$). Concretely, we initiate it by random generation of its decomposition coefficients. To probe the function's behavior under various orientations, we applied 100 random rotations to it, producing a collection of rotated copies. For each of these rotated versions, we found its sampled maximum over the SO(3) space with the sampling size K. Aggregating these maxima across all rotations allowed us to determine their standard deviation.

Figure 2 shows the normalized standard deviation (relative to the true maximum of the initial function) as a function of the linear sampling size K. Even for relatively small values of $K = L$, the ratio between the standard deviation of the maxima and the true maximum hovers around 10^{-2}. This implies that the deviation of the sampling maximum from the true maximum remains minimal, underscoring the reliability of our orientation-independent pattern detection across varying sampling resolutions.

4.2 Experiments on the CATH Dataset

For our first experiment, we chose a volumetric voxelized dataset from [30] composed of 3D protein conformations classified according to the CATH hierarcy. The CATH Protein Structure Classification Database provides a hierarchical classification of 3D conformations of protein domains, i.e., compact self-stabilizing protein regions that folds independently [15]. The dataset considers the "architecture" level in the CATH hierarchy, version 4.2 (see `http://cathdb.info/browse/tree`). It focuses on "architectures" with a minimum of 700 members, producing ten distinct classes. All classes are represented by the same number of proteins. Each protein in the dataset is described by its alpha-carbon positions that are placed on the volumetric grid of the linear size 50. The dataset is available at `https://github.com/wouterboomsma/cath_datasets` [30]. For benchmarking, the authors of the dataset also provide a 10-fold split ensuring the variability of proteins from different splits.

For the experiment, we constructed three architectures (ILPONet, ILPONet-small, and ILPONet-tiny) with different numbers of trainable parameters, and also tested the two types of pooling operations. ILPONet, ILPONet-small, and ILPONet-tiny replicate the architecture of ResNet-34 [8], but they impliment the novel convolution operation with 4, 8, and 16 times fewer channels on each layer, respectively. We conducted experiments for two types of orientation pooling with $K = 4$ for the softmax version, and $K = 7$ for the hardmax version.

We compared the performance of ILPO-Net (our method) with two baselines: ResNet-34 and its equivariant version, ResNet-34 with Steerable filters, whose performance was demonstrated in [30] where the dataset was introduced. Table 1 lists the accuracy (**ACC**) and the number of parameters(**# of params**) of different tested methods. Since the classes in the dataset are balanced, we can use accuracy as the sole metric to evaluate the precision of predictions.

Table 1. Performance comparison of various methods on the CATH dataset.

Method	**ACC**	**# of params**
ResNet-34 [8]	0.61	15M
Steerable ResNet-34 [30]	0.66	150K
ILPONet-34(hardmax)	**0.74**	1M
ILPONet-34(softmax)	0.74	1M
ILPONet-34(hardmax)-small	0.73	258k
ILPONet-34(softmax)-small	0.72	258k
ILPONet-34(hardmax)-tiny	0.68	65k
ILPONet-34(softmax)-tiny	0.70	65k

As shown in Table 1, all versions of ILPO-Net outperform both baselines on the CATH dataset. Furthermore, when comparing the number of parameters, even the smallest variant of ILPO-Net achieves a better accuracy, while having substantially fewer parameters than the equivariant baseline, Steerable Network.

Technical details: We used the first 7 splits for training, 1 for validation, and 2 for testing following the protocol of [30]. We trained our models for 100 epochs with the Adam optimizer [13] and an exponential learning rate decay of 0.94 per epoch starting after an initial burn-in phase of 40 epochs. We used a 0.01 dropout rate, and $L1$ and $L2$ regularization values of 10^{-7}. For the final model, we chose the epoch where the validation accuracy was the highest. Table 1 shows the performance on the test data. We based our experiments on the framework provided by [30] in their **se3cnn** repository. We introduced our ILPO operator into the provided setup for training and evaluation.

4.3 Experiments on MedMNIST Datasets

For the second experiment, we selected MedMNIST v2, a vast MNIST-like collection of standardized biomedical images [34]. This collection covers 12 datasets for 2D and 6 datasets for 3D images. Preprocessing reduced all images into the standard size of 28×28 for 2D and $28 \times 28 \times 28$ for 3D, each with its corresponding classification labels. MedMNIST v2 data are supplied with tasks ranging from binary/multi-class classification to ordinal regression and multi-label classification. The collection, in total, consists of 708,069 2D images and 9,998 3D images. For this study, we focused only on the 3D datasets of MedMNIST v2.

As the baseline, we used the same models as the authors of the collection tested on 3D datasets. These are multiple versions of ResNet [8] with 2.5D/3D/ACS [33] convolutions and open-source AutoML tools, auto-sklearn [7], AutoKeras [10], FPVT [20], and Moving Frame Net [26]. As in the previous experiment, we constructed and trained multiple architectures (ILPONet, and ILPONet-small) of different size with two versions of the orientation pooling operation. They repeat the sequence of layers in ResNet-18 and ResNet-50 but they do not reduce the size of the spatial input dimension throughout the network.

The models ILPONet-small and ILPONet keep 4 and 8 feature channels, respectively, throughout the network. We tested these architectures for both soft- and hardmax orientation pooling strategies. Table 2 lists the performance of our models compared to the baselines. Here, the classes are not balanced. Therefore, the accuracy (**ACC**) cannot be the only indicator of the prediction precision, and we also consider AUC-ROC(**AUC**) that is more revealing. We can see that ILPOResNet models, even with a substantially reduced number of parameters, demonstrate competitive or superior performance compared to traditional methods on the 3D datasets of MedMNIST v2.

Technical details: For each dataset, we used the training-validation-test split provided by [34]. We utilized the Adam optimizer with an initial learning rate of 0.0005 and trained the model for 100 epochs, delaying the learning rate by 0.1 after 50 and 75 epochs. The dropout rate was 0.01. To test the model, we chose the epoch corresponding to the best **AUC** on the validation set. We based our experiments on the framework provided by [34] in their **MedMNIST** repository. We introduced our ILPO operator into their setup for training and evaluation.

Table 2. Comparison of different methods on MedMNIST's 3D datasets. (*) For these methods, the number of parameters is unknown. The best accuracies (ACC) and ROC-areas under curve (AUC) are highlighted in bold.

Methods	**# of params**	**Organ**		**Nodule**		**Fracture**		**Adrenal**		**Vessel**		**Synapse**	
		AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC
ResNet-18 [8] + 2.5D[33]	11M	0.977	0.788	0.838	0.835	0.587	0.451	0.718	0.772	0.748	0.846	0.634	0.696
ResNet-18 [8]+ 3D[33]	33M	**0.996**	**0.907**	0.863	0.844	0.712	0.508	0.827	0.721	0.874	0.877	0.820	0.745
ResNet-18 [8]+ ACS[33]	11M	0.994	0.900	0.873	0.847	0.714	0.497	0.839	0.754	0.930	0.928	0.705	0.722
ResNet-50 [8]+ 2.5D[33]	15M	0.974	0.769	0.835	0.848	0.552	0.397	0.732	0.763	0.751	0.877	0.669	0.735
ResNet-50 [8]+ 3D[33]	44M	0.994	0.883	0.875	0.847	0.725	0.494	0.828	0.745	0.907	0.918	0.851	0.795
ResNet-50 [8]+ ACS[33]	15M	0.994	0.889	0.886	0.841	0.750	0.517	0.828	0.758	0.912	0.858	0.719	0.709
auto-sklearn* [7]	-	0.977	0.814	**0.914**	**0.874**	0.628	0.453	0.828	0.802	0.910	0.915	0.631	0.730
AutoKeras* [10]	-	0.979	0.804	0.844	0.834	0.642	0.458	0.804	0.705	0.773	0.894	0.538	0.724
FPVT* [20]	-	0.923	0.800	0.814	0.822	0.640	0.438	0.801	0.704	0.770	0.888	0.530	0.712
SE3MovFrNet * [26]	-	-	0.745	-	0.871	-	0.615	-	0.815	-	**0.953**	-	0.896
ILPOResNet-18(softmax)-small	7k	0.960	0.631	0.887	0.848	0.791	0.579	0.897	0.805	0.815	0.838	0.804	0.517
ILPOResNet-18(hardmax)-small	7k	0.951	0.600	0.906	0.861	**0.808**	**0.642**	0.870	0.792	0.925	0.908	0.825	0.750
ILPOResNet-18(softmax)	29k	0.967	0.716	0.894	0.871	0.761	0.558	**0.910**	**0.856**	0.908	0.919	0.836	0.815
ILPOResNet-18(hardmax)	29k	0.971	0.705	0.900	**0.874**	0.773	0.580	0.897	0.846	0.927	0.908	0.800	0.767
ILPOResNet-50(softmax)-small	10k	0.979	0.757	0.902	0.865	0.772	0.558	0.864	0.745	0.864	0.890	0.880	0.844
ILPOResNet-50(hardmax)-small	10k	0.981	0.780	0.887	0.861	0.768	0.571	0.841	0.792	**0.937**	0.901	0.861	0.784
ILPOResNet-50(softmax)	38k	0.992	0.879	0.912	0.871	0.767	0.608	0.869	0.809	0.829	0.851	**0.940**	**0.923**
ILPOResNet-50(hardmax)	38k	0.975	0.754	0.911	0.839	0.769	0.521	0.893	0.842	0.902	0.885	0.885	0.858

4.4 Filter demonstration

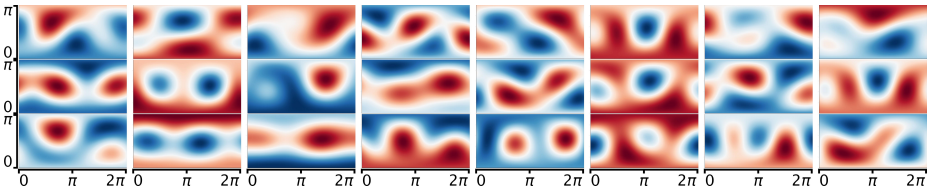

Fig. 3. Visualization of filters from the 1st ILPO layer of ILPONet-50. Each column corresponds to different output channels, with rows indicating different radii and input channels. Given that the first ILPO layer only has one input channel, only three projections (radii) are shown in each column. x and y axes correspond to the azimuthal and polar angles, correspondingly. The filters' values are shown in the Mercator projection. The red color corresponds to the positive values, and the blue color to the negative ones.

For a deeper understanding of our models, it is useful to delve into the visualizations of their filters. Of the numerous experiments conducted, we opted to focus on the MedMNIST experiments, primarily due to the smaller size of the trained models (in terms of parameter count). Within the MedMNIST collection, we chose the Synapse dataset because of its more sophisticated and variable patterns and analyzed the filters from the top-performing ILPONet-50 model with the softmax orientation pooling. This architecture employs ILPO convolutional layers, each having a filter size of $L = 3$. Here, we demonstrate filters from the first and the last ILPO layers. Depending on the radius (r), these filters could represent a single point (for $r = 0$) or spheres for other radii values. We use

the Mercator projection to show values on the filters' spheres for $r > 0$ in two spherical angles, azimuthal and polar.

Figure 3 shows the first ILPO layer. The layer has a single input channel. Different rows correspond to different radii($r = 1, \sqrt{2}, \sqrt{3}$), whereas each column corresponds to a different output channel. Appendix D also shows the last ILPO layer. These figures demonstrate a variety of memorized patterns. We can see no spatial symmetry in the filters and that the presented model can learn filters of arbitrary shape. Interestingly, we cannot spot a clear difference between the filters of the first and the last layers.

5 Discussion and Conclusion

In real-world scenarios, data augmentation is commonly employed to achieve rotational and other invariances of DL models. While this method may significantly increase the dataset's size and the number of parameters, it can also limit the expressivity and explainability of the obtained models. Using invariant methods by design is a valid alternative that ensures consistent neural network performance. The filter representation introduced here can also be employed in an equivariant architecture. It will lead to a higher complexity of operations and an increased number of parameters but may give better expressiveness to the final model.

To conclude, we proposed the ILPO-NET approach that efficiently manages arbitrarily shaped patterns, providing inherent invariance to local spatial pattern orientations through the novel convolution operation. When tested against several volumetric datasets, ILPO-Net demonstrated state-of-the-art performance with a remarkable reduction in parameter counts. Its potential extends beyond the tested cases, with promising applications across multiple disciplines.

Acknowledgments. We wish to express our profound gratitude to Sylvain Meignen, associate professor at Grenoble-INP, for his invaluable advice. We would also like to thank Jérôme Malick, Laurence Wazné, and Kliment Olechnovic for their support during the study. This work was partially supported by MIAI@Grenoble Alpes (ANR-19-P3IA-0003).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Andrearczyk, V., Fageot, J., Oreiller, V., Montet, X., Depeursinge, A.: Local rotation invariance in 3D CNNs. Medical Image Analysis **65**, 101756 (Oct 2020). https://doi.org/10.1016/j.media.2020.101756, https://doi.org/10.1016/j.media.2020.101756
2. Bekkers, E.J., Lafarge, M.W., Veta, M., Eppenhof, K.A., Pluim, J.P., Duits, R.: Roto-translation covariant convolutional networks for medical image analysis. In: Medical Image Computing and Computer Assisted Intervention–MICCAI 2018:

21st International Conference, Granada, Spain, September 16-20, 2018, Proceedings, Part I. pp. 440–448. Springer (2018)
3. Cohen, T., Weiler, M., Kicanaoglu, B., Welling, M.: Gauge equivariant convolutional networks and the icosahedral CNN. In: International conference on Machine learning. pp. 1321–1330. PMLR (2019)
4. Cohen, T., Welling, M.: Group equivariant convolutional networks. In: International conference on machine learning. pp. 2990–2999. PMLR (2016)
5. Cohen, T.S., Geiger, M., Köhler, J., Welling, M.: Spherical CNNs. In: International Conference on Learning Representations (2018)
6. Dehmamy, N., Walters, R., Liu, Y., Wang, D., Yu, R.: Automatic symmetry discovery with lie algebra convolutional network. Advances in Neural Information Processing Systems **34**, 2503–2515 (2021)
7. Feurer, M., Klein, A., Eggensperger, K., Springenberg, J.T., Blum, M., Hutter, F.: Auto-sklearn: Efficient and robust automated machine learning. In: Automated Machine Learning, pp. 113–134. Springer International Publishing (2019). https://doi.org/10.1007/978-3-030-05318-5_6, https://doi.org/10.1007/978-3-030-05318-5_6
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE (Jun 2016). https://doi.org/10.1109/cvpr.2016.90, https://doi.org/10.1109/cvpr.2016.90
9. Igashov, I., Pavlichenko, N., Grudinin, S.: Spherical convolutions on molecular graphs for protein model quality assessment. Machine Learning: Science and Technology **2**(4), 045005 (2021)
10. Jin, H., Song, Q., Hu, X.: Auto-keras: An efficient neural architecture search system. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM (Jul 2019). https://doi.org/10.1145/3292500.3330648, https://doi.org/10.1145/3292500.3330648
11. Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., Bridgland, A., Meyer, C., Kohl, S.A.A., Ballard, A.J., Cowie, A., Romera-Paredes, B., Nikolov, S., Jain, R., Adler, J., Back, T., Petersen, S., Reiman, D., Clancy, E., Zielinski, M., Steinegger, M., Pacholska, M., Berghammer, T., Bodenstein, S., Silver, D., Vinyals, O., Senior, A.W., Kavukcuoglu, K., Kohli, P., Hassabis, D.: Highly accurate protein structure prediction with AlphaFold. Nature **596**(7873), 583–589 (Jul 2021). https://doi.org/10.1038/s41586-021-03819-2, https://doi.org/10.1038/s41586-021-03819-2
12. Khalid, Z., Durrani, S., Kennedy, R.A., Wiaux, Y., McEwen, J.D.: Gauss-legendre sampling on the rotation group. IEEE Signal Processing Letters **23**(2), 207–211 (Feb 2016). https://doi.org/10.1109/lsp.2015.2503295, https://doi.org/10.1109/lsp.2015.2503295
13. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
14. Knigge, D.M., Romero, D.W., Bekkers, E.J.: Exploiting redundancy: Separable group convolutional networks on lie groups. In: International Conference on Machine Learning. pp. 11359–11386. PMLR (2022)
15. Knudsen, M., Wiuf, C.: The CATH database. Human Genomics **4**(3), 207 (2010). https://doi.org/10.1186/1479-7364-4-3-207, https://doi.org/10.1186/1479-7364-4-3-207
16. Kondor, R.: N-body networks: a covariant hierarchical neural network architecture for learning atomic potentials. ArXiv **abs/1803.01588** (2018), https://api.semanticscholar.org/CorpusID:3665386

17. Kondor, R., Lin, Z., Trivedi, S.: Clebsch–gordan nets: a fully fourier space spherical convolutional neural network. Advances in Neural Information Processing Systems **31** (2018)
18. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems **25** (2012)
19. Liu, C., Ruhe, D., Eijkelboom, F., Forré, P.: Clifford group equivariant simplicial message passing networks. arXiv preprint arXiv:2402.10011 (2024)
20. Liu, J., Li, Y., Cao, G., Liu, Y., Cao, W.: Feature pyramid vision transformer for MedMNIST classification decathlon. In: 2022 International Joint Conference on Neural Networks (IJCNN). IEEE (Jul 2022). https://doi.org/10.1109/ijcnn55064.2022.9892282, https://doi.org/10.1109/ijcnn55064.2022.9892282
21. Liu, R., Lauze, F., Bekkers, E., Erleben, K., Darkner, S.: Group convolutional neural networks for dwi segmentation. In: Geometric Deep Learning in Medical Image Analysis. pp. 96–106. PMLR (2022)
22. Pagès, G., Charmettant, B., Grudinin, S.: Protein model quality assessment using 3D oriented convolutional neural networks. Bioinformatics **35**(18), 3313–3319 (Feb 2019). https://doi.org/10.1093/bioinformatics/btz122, https://doi.org/10.1093/bioinformatics/btz122
23. Romero, D., Bekkers, E., Tomczak, J., Hoogendoorn, M.: Attentive group equivariant convolutional networks. In: International Conference on Machine Learning. pp. 8188–8199. PMLR (2020)
24. Roth, C., MacDonald, A.H.: Group convolutional neural networks improve quantum state accuracy. arXiv preprint arXiv:2104.05085 (2021)
25. Ruhe, D., Brandstetter, J., Forré, P.: Clifford group equivariant neural networks. arXiv preprint arXiv:2305.11141 (2023)
26. Sangalli, M., Blusseau, S., Velasco-Forero, S., Angulo, J.: Moving frame net: Se (3)-equivariant network for volumes. In: NeurIPS Workshop on Symmetry and Geometry in Neural Representations. pp. 81–97. PMLR (2023)
27. Satorras, V.G., Hoogeboom, E., Welling, M.: E (n) equivariant graph neural networks. In: International conference on machine learning. pp. 9323–9332. PMLR (2021)
28. Thomas, N., Smidt, T.E., Kearnes, S.M., Yang, L., Li, L., Kohlhoff, K., Riley, P.F.: Tensor field networks: Rotation- and translation-equivariant neural networks for 3D point clouds. ArXiv **abs/1802.08219** (2018), https://api.semanticscholar.org/CorpusID:3457605
29. Wang, B., Lei, Y., Tian, S., Wang, T., Liu, Y., Patel, P., Jani, A.B., Mao, H., Curran, W.J., Liu, T., et al.: Deeply supervised 3D fully convolutional networks with group dilated convolution for automatic mri prostate segmentation. Medical physics **46**(4), 1707–1718 (2019)
30. Weiler, M., Geiger, M., Welling, M., Boomsma, W., Cohen, T.S.: 3D steerable CNNs: Learning rotationally equivariant features in volumetric data. Advances in Neural Information Processing Systems **31** (2018)
31. Winkels, M., Cohen, T.S.: 3D G-CNNs for pulmonary nodule detection. In: Medical Imaging with Deep Learning (2022)
32. Worrall, D., Brostow, G.: CubeNet: Equivariance to 3D rotation and translation. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 567–584 (2018)
33. Yang, J., Huang, X., He, Y., Xu, J., Yang, C., Xu, G., Ni, B.: Reinventing 2D convolutions for 3D images. IEEE Journal of Biomedical and Health Informat-

ics **25**(8), 3009–3018 (Aug 2021). https://doi.org/10.1109/jbhi.2021.3049452, https://doi.org/10.1109/jbhi.2021.3049452
34. Yang, J., Shi, R., Wei, D., Liu, Z., Zhao, L., Ke, B., Pfister, H., Ni, B.: MedMNIST v2 - a large-scale lightweight benchmark for 2D and 3D biomedical image classification. Scientific Data **10**(1) (Jan 2023). https://doi.org/10.1038/s41597-022-01721-8, https://doi.org/10.1038/s41597-022-01721-8
35. Zhdanov, M., Ruhe, D., Weiler, M., Lucic, A., Brandstetter, J., Forré, P.: Clifford-steerable convolutional neural networks. arXiv preprint arXiv:2402.14730 (2024)
36. Zhemchuzhnikov, D., Igashov, I., Grudinin, S.: 6DCNN with roto-translational convolution filters for volumetric data processing. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 36, pp. 4707–4715 (2022)

Appendix

A Limitation of expressiveness in Steerable Networks

Convolution operations, foundational to modern Convolutional Neural Networks (CNNs), serve as a mechanism for detecting patterns in input data. In the traditional convolution, higher activation values in the feature map indicate regions in the input where there is a significant match with the convolutional filter, thereby signaling the presence of a targeted pattern.

Let us consider how this mechanism works in convolutions with steerable filters [30]. The steerable filter that maps between irreducible features ($i \to l$) is defined as:

$$\kappa_{il}(\boldsymbol{r}) = \sum_{L=|i-l|}^{i+l} \sum_{n=0}^{N-1} w_{il,Ln} \kappa_{il,Ln}(\boldsymbol{r}), \tag{22}$$

where $\kappa_{il}(\boldsymbol{r}) : \mathbb{R}^3 \to \mathbb{R}^{(2i+1)(2l+1)}$. Here, $w_{il,Ln}$ are learnable weights and $\kappa_{il,Ln}$ are basis functions given by:

$$\kappa_{il,Ln}(\boldsymbol{r}) = Q^{ilL} \eta_{Ln}(\boldsymbol{r}), \tag{23}$$

where $Q^{ilL} \in \mathbb{R}^{(2i+1)(2l+1)\times(2L+1)}$ is the 3-dimensional tensor with Clebsch-Gordon coefficients and

$$\eta_{Ln}(\boldsymbol{r}) = \phi_n(r) Y_L(\Omega_r), \tag{24}$$

$\eta_{Ln}(\boldsymbol{r}) : \mathbb{R}^3 \to \mathbb{R}^{(2L+1)}$ and $Y_L(\Omega_r)$ is a vector with spherical harmonics of degree L. Functions $\phi_n(n = 0, ..., N-1)$ form a radial basis. For a scalar field as input data and considering the special case $l = 0$, the filter reduces to

$$\kappa_{i0}(\boldsymbol{r}) = \sum_{n=0}^{N-1} w_{i0,in} \phi_n(r) Y_i(\Omega_r), \tag{25}$$

where $\kappa_{i0}(\boldsymbol{r}) : \mathbb{R}^3 \to \mathbb{R}^{(2i+1)\times 1}$.

Let us apply the convolution to the following input function,

$$f(\boldsymbol{r}) = \sum_{i=0}^{L_{\max}} \sum_{m=-i}^{i} \sum_{n=0}^{N} f_{in}^m \phi_n(r) Y_i^m(\Omega_r), \tag{26}$$

where indices i, m correspond to the angular decomposition and n is a radial index. Without loss of generality for the final conclusion, let us consider a special case when coefficients f_{in}^m can be expressed as a product: $f_{in}^m = f_i^m q_{in}$. We also assume that the pattern presented by this function is localised and the function is defined in a cube. The filter $\kappa_{i0}(\boldsymbol{r})$ is localised in a cube of the same size. If we use the integral formulation, the result of the convolution operation at the center of the pattern will be:

$$h_i^m = \int_0^\infty \int_{S^2} f(\boldsymbol{r}) [\kappa_{i0}(\boldsymbol{r})]_{m0} d\Omega_r r^2 dr = f_i^m \sum_{n=0}^{N-1} w_{i0,in} q_{in}. \tag{27}$$

Then, according to the logic of the convolution layer, a nonlinear operator is applied to the convolution result, which zeros the low signal level. Let us consider two types of nonlinearities used in 3D Steerable networks: gated- and norm-nonlinearity. In these operators, the high-degree output of the convolution result $(h_i^m, i > 0)$ is multiplied with $\sigma(h_0^0)$ and $\sigma(\|h_i\|)$, respectively, where σ is an activation function and $\|h_i\| = \sqrt{\sum_{m=-i}^{i}(h_i^m)^2} = |\sum_{n=0}^{N-1} w_{i0,in}q_{in}|\sqrt{\sum_{m=-i}^{i}(f_i^m)^2}$ is the norm of the ith-degree coefficients . In the first case, the gated non-linearity does not distinguish patterns of different shapes if they have the same decomposition coefficients of the 0th degree(f_{0n}^0). The norm non-linearity brings more expressiveness for representations of the 1st-degree because if two sets of representations, $\{f_1^{-1}, f_1^0, f_1^1\}$ and $\{[f']_1^{-1}, [f']_1^0, [f']_1^1\}$, have equal norms ($\|f_1\| = \|[f']_1\|$), then f_1 can be retrieved from $[f']_1$ by a rotation or, in other words, they represent the same shape . However, this rule does not work for higher degrees $(i \geq 2)$. For example, representations of the 2nd-degree $f_2 = \{1, 0, 0, 0, 0\}$ and $[f']_2 = \{0, 0, 1, 0, 0\}$ represent different shapes but have equal norms.

Accordingly, a single layer cannot cope with the recognition of an arbitrary pattern in the input data. Thus, the recognition task moves to the subsequent layers. However, on the second layer, there is an exchange between the voxel of the feature map where h_i is stored and other voxels that contain not only the pattern information but also the pattern's neighbors information. Therefore, the result of the central pattern recognition will not be unique but depends on the pattern neighbors.

B Limitation of summing up over rotations

Averaging (or summing) of a function in $3D$ annihilates angular dependencies of a filter. Let us consider a filter defined by a function $g(\boldsymbol{r})$:

$$g(\boldsymbol{r}) = \sum_{l=0}^{L-1} g_l^k(r) Y_l^k(\Omega_r), \tag{28}$$

where (r, Ω_r) are the radial and the angular components of the vector $\boldsymbol{r}$, and g_l^k are the spherical harmonic expansion coefficients of a function $g(\boldsymbol{r})$. We then rotate this function by $\mathcal{R} \in \mathrm{SO}(3)$ and convolve with $f(\boldsymbol{r})$:

$$h(\boldsymbol{\Delta}, \mathcal{R}) = \int_{\mathbb{R}^3} f(\boldsymbol{\Delta} - \boldsymbol{r}) g(\boldsymbol{r}) d\boldsymbol{r}. \tag{29}$$

If we integrate this result over all rotations in SO(3), which is approximately equal to summing the function over a finite set of (equally distributed) rotations, we obtain

$$h(\boldsymbol{\Delta}) = \int_{\mathrm{SO}(3)} h(\boldsymbol{\Delta}, \mathcal{R}) d\mathcal{R} = 8\pi^2 \int_{\mathbb{R}^3} f(\boldsymbol{\Delta} - \boldsymbol{r}) g_0^0(r) d\boldsymbol{r}, \tag{30}$$

where $g_0^0(r)$ are the zero-order expansion coefficients that equal to the mean value of the integrated function over the domain. Thus, we can conclude that summing

over all rotations in SO(3) of the result of the convolution with an arbitrary filter is equivalent to a convolution with a radially-symmetric filter. On the contrary, Eq. 16 allows us to keep the dependency of the convolution result on the filter's orientation.

Theorem 3 provides an upper bound for the error of the sampled maximum. The softmax is limited by the hard maximum value for the continuous and discrete cases, consequently the sampled softmax error is also bounded. We also deduced an *empirical* relationship between the error and parameters L and K for both operations. For example, for $L = 3$ the error of the softmax approximation follows the relation $\epsilon = 4K^{-3}$. Therefore, for $\epsilon = 0.1, 0.05$ or 0.01 we need to consider $K = 4, 5$ or 7, respectively. The error of the sampling hardmax is approximately $2.75K^{-2}$ if $L = 3$. It means that $K = 7, 9$ or 30 will give $\epsilon = 0.1, 0.05$ or 0.01 respectively.

The discrete calculation of the hard maximumum does not differ from the continuous case. The discrete form of the soft maximum operation has the following expression:

$$\text{softmax}_{\mathcal{R}} f(x_i, y_j, z_k, \mathcal{R}) = \frac{\sum_{q,r,s} w_r \text{relu}(h(x_i, y_j, z_k, \alpha_q, \beta_r, \gamma_s))^2}{\sum_{q,r,s} w_r \text{relu}(h(x_i, y_j, z_k, \alpha_q, \beta_r, \gamma_s))}, \tag{31}$$

where w_r are the Gauss–Legendre quadrature weights.

C Upper bound of the sampling maximum error

Lemma 1. *Let $Y_l^k(\theta, \phi)$ be the spherical harmonic function of degree l and order k. Then, the Lipschitz constant L of $Y_l^k(\theta, \phi)$ is bounded by:*

$$L \leq \sqrt{l(l+1)}$$

Proof. Given the following differential relations:

$$\frac{\delta Y_l^k(\theta, \phi)}{\delta \theta} = k Y_l^{-k}(\theta, \phi) \tag{32}$$

$$\frac{\delta Y_l^k(\theta, \phi)}{\delta \phi} = Y_l^k(\theta, \phi) l \cot(\phi) - Y_{l-1}^k(\theta, \phi) \frac{\sqrt{(l-k)(l+k)}}{\sin(\phi)} \frac{2l+1}{2l-1}, \tag{33}$$

we can obtain the expression for the gradient of $Y_l^k(\theta, \phi)$ as:

$$\nabla Y_l^k = \left(\frac{\delta Y_l^k}{\delta \theta}, \frac{\delta Y_l^k}{\delta \phi} \right). \tag{34}$$

To determine the Lipschitz constant, we find the maximum magnitude of the gradient over the function's domain. Using the provided differential relations, the squared magnitude of the gradient is:

$$\|\nabla Y_l^k\|^2 = \left(k Y_l^{-k}\right)^2 + \left(Y_l^k l \cot(\phi) - Y_{l-1}^k \frac{\sqrt{(l-k)(l+k)}}{\sin(\phi)} \frac{2l+1}{2l-1} \right)^2. \tag{35}$$

Given that $\|k\| \leq l$, the term k^2 is bounded by l^2. The dominant term from the second expression is $l\cot(\phi)$, which in the worst case is proportional to l^2. Thus, the Lipschitz constant is bounded by the square root of the maximum term from the gradient's squared magnitude. This gives:

$$L \leq \sqrt{l(l+1)}. \tag{36}$$

Theorem 1. *The Lipschitz constant L_D of the Wigner matrix element $D^l_{k_1k_2}(\mathcal{R})$ is bounded by:*

$$L_D \leq 4\pi\sqrt{l(l+1)}$$

Proof. First, recall the expression for the Wigner matrix element:

$$D^l_{k_1k_2}(\mathcal{R}) = \int_{SO(2)} Y_l^{k_1}(\mathcal{R}x) Y_l^{k_2}(x)\, dx, \tag{37}$$

where $x = x(\theta, \phi)$ is a solid angle, and $\mathcal{R}$ is a rotation in SO(3). To determine the Lipschitz constant for the Wigner matrix element, we find the magnitude of its gradient with respect to $\mathcal{R}$. Using the chain rule:

$$\frac{\partial Y_l^k(\mathcal{R}x)}{\partial \mathcal{R}} = \frac{\partial Y_l^k(x)}{\partial x}(\mathcal{R}x)\frac{\partial(\mathcal{R}x)}{\partial \mathcal{R}}. \tag{38}$$

Given the lemma above, we know that the Lipschitz constant L for the spherical harmonic $Y_l^k(\theta, \phi)$ is bounded by $\sqrt{l(l+1)}$. Thus,

$$\max \|\frac{\partial D^l_{k_1k_2}(\mathcal{R})}{\partial \mathcal{R}}\| \leq 4\pi\sqrt{l(l+1)} \max Y_l^{k_2} \leq 4\pi\sqrt{l(l+1)}. \tag{39}$$

Theorem 2. *Let $f(\mathcal{R})$ be a function in SO(3) whose maximum degree of Wigner matrices decomposition is $L-1$ and whose 2-norm is C. Then, the Lipschitz constant L_f of f is bounded by:*

$$L_f \leq 4\frac{C}{\sqrt{3}} L^{\frac{5}{2}}$$

Proof. Given the decomposition of the function f in terms of Wigner matrices,

$$f(\mathcal{R}) = \sum_{l=0}^{L-1} \sum_{k_1=-l}^{l} \sum_{k_2=-l}^{l} f^l_{k_1k_2} D^l_{k_1k_2}(\mathcal{R}), \tag{40}$$

we also have the expression for the 2-norm squared of f,

$$\|f\|_2^2 = \sum_{l=0}^{L-1} \sum_{k_1=-l}^{l} \sum_{k_2=-l}^{l} \frac{8\pi^2}{2l+1} \|f^l_{k_1k_2}\|^2 = C^2. \tag{41}$$

Knowing that

$$(\sum_{l=0}^{L-1} \sum_{k_1=-l}^{l} \sum_{k_2=-l}^{l} |f^l_{k_1k_2}|)^2 \leq \frac{(4L^3 - L)}{3} (\sum_{l=0}^{L-1} \sum_{k_1=-l}^{l} \sum_{k_2=-l}^{l} \|f^l_{k_1k_2}\|^2), \tag{42}$$

we deduce:

$$\frac{8\pi^2}{2L-1}\left(\sum_{l=0}^{L-1}\sum_{k_1=-l}^{l}\sum_{k_2=-l}^{l}|f_{k_1k_2}^{l}|\right)^2 \leq \frac{(4L^3-L)}{3}C^2. \tag{43}$$

From the previous expression it follows that:

$$\sum_{l=0}^{L-1}\sum_{k_1=-l}^{l}\sum_{k_2=-l}^{l}\|f_{k_1k_2}^{l}\| \leq \sqrt{C^2\frac{(4L^3-L)}{3}\frac{2L-1}{8\pi^2}}. \tag{44}$$

Considering that the Lipschitz constant for $D_{k_1k_2}^{l}(\mathcal{R})$ is $4\pi\sqrt{l(l+1)}$, the Lipschitz constant for $f(\mathcal{R})$ is bounded by the product of the maximum Lipschitz constant for the Wigner matrices and the maximum magnitude of the coefficients. Thus,

$$L_f \leq 4\pi\sqrt{C\frac{(4L^3-L)}{3}\frac{2L-1}{8\pi^2}}\sqrt{(L-1)L} \leq 4\frac{C}{\sqrt{3}}L^{\frac{5}{2}}. \tag{45}$$

This concludes the proof.

Theorem 3. *Let the function $f(\mathcal{R})$ be defined in SO(3) with its maximum degree of Wigner matrices decomposition being $L-1$:*

$$f(\mathcal{R}) = \sum_{l=0}^{L-1}\sum_{m_1=-l}^{l}\sum_{m_2=-l}^{l} f_{m_1m_2}^{l} D_{m_1m_2}^{l}(\mathcal{R}),$$

with the 2-norm of this function $C<\infty$. Given a sampling $\alpha_{k_1} = k_1\frac{2\pi}{K}, k_1 = 0,...,K$, $\beta_{k_2} = \arccos(x_{k_2})$ where x_i are Gauss-Legendre quadrature points of K, and $\gamma_{k_3} = k_3\frac{2\pi}{K}, k_3 = 0,...,K$, if $K > K_0$ where $K_0 = \frac{8\pi L^{\frac{5}{2}}C/\sqrt{3}}{\epsilon}$, then the discrepancy between the sampled maximum and the true maximum of f over its domain is smaller than ϵ.

Proof. Using the Lipschitz constant from Theorem 2, we get:

$$|f(\mathbf{u}) - f(\mathbf{v})| \leq 4\frac{C}{\sqrt{3}}L^{\frac{5}{2}}\|\mathbf{u}-\mathbf{v}\|. \tag{46}$$

The largest difference in successive sampled points in α and γ will be :

$$\|\mathbf{u}_{\text{successive}} - \mathbf{v}_{\text{successive}}\| = \frac{2\pi}{K}. \tag{47}$$

For the sampling in β we obtain the same relation,

$$\max_i |\beta_{i+1} - \beta_i| \leq \frac{2\pi}{K}. \tag{48}$$

Using the Lipschitz property and combining the discrepancies, we deduce:

$$|f(\mathbf{u}_{\text{successive}}) - f(\mathbf{v}_{\text{successive}})| \leq 4\frac{C}{\sqrt{3}}L^{\frac{5}{2}}\frac{2\pi}{K}. \tag{49}$$

For the above discrepancy to be smaller than ϵ, we must require:

$$K > \frac{8\pi L^{\frac{5}{2}}C/\sqrt{3}}{\epsilon}. \tag{50}$$

Thus, the smallest such a value for K is $K_0 = \frac{8\pi L^{\frac{5}{2}}C/\sqrt{3}}{\epsilon}$.

D Filter Demonstration

Figure 4 visualizes the last, 17th ILPO layer from the ILPONet-50 model trained on the Synapse dataset of the MedMNIST collection. This layer has multiple input channels, therefore we split each column into triplets corresponding to different input channels.

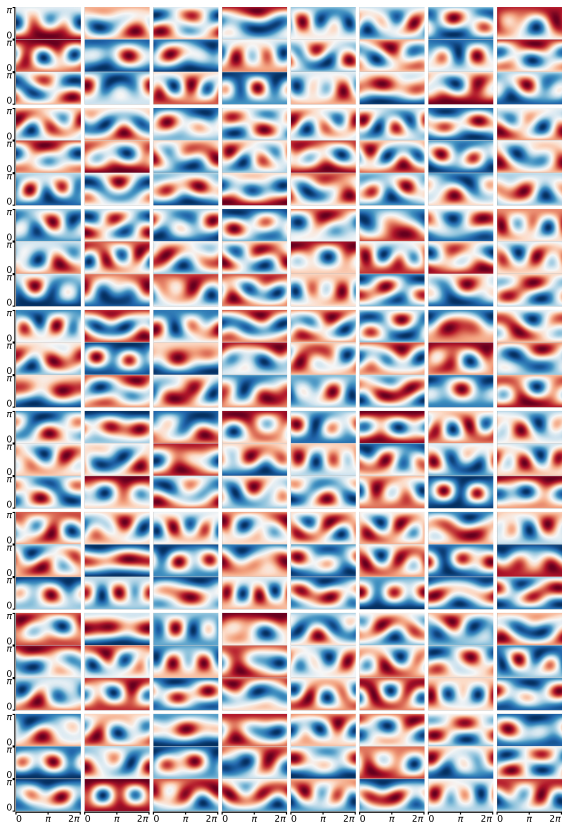

Fig. 4. Visualization of filters from the last, 17th ILFO layer of ILFONet-50. Each column in the illustration represents a triplet corresponding to three different radii in the filter. Different triplets relate to different input channels, reflecting the complexity and feature extraction capabilities of deeper layers in the network. x and y axes correspond to the azimuthal and polar angles, correspondingly. The filters' values are shown in the Mercator projection. The red color corresponds to the positive values, and the blue color to the negative ones.