Equivariant Neural Networks of Tensors on SO(3) for Weather Prediction

Francesco Ballerin

francesco.ballerin@uib.no

Joint work with **Erlend Grong** and **Nello Blaser**

Abstract

Geometric Deep Learning (GDL) is a field that extends traditional deep learning methods to data with an underlying geometric structure, such as graphs and manifolds. Neural Networks, and machine learning models in general, are sensitive to how data is organized and stored. **GDL networks are natively designed with mechanisms to handle non-Euclidean data** and force either **equivariance** or **invariance**, depending on the type of problem. This means that the result of a prediction does not depend on the symmetries of the underlining geometrical space: rotating a picture of a cat does not produce a picture of a dog and changing atlas of a manifold does not change the manifold. This tackles some of the problems of the **black-box nature** of Neural Networks forcing some useful mathematical **constraints**. Our work explores the ideas of GDL applied to tensors, with an application to meteorological data prediction.

Symmetries

When dealing with data in a digital setting we have to make a conscious choice on how to **encode** it in a computer. If we take one image and we flip it or rotate it, the underlying image is the same but the way the data is stored and processed is different. This difference in how the data is encoded can **affect the prediction** of a machine-learning model. These transformations, which preserve the underlying data but encode it in different ways, take the name of **symmetries**. A naive solution to the problem of symmetries lies within **data augmentation**: we create multiple versions of a datapoint in order to consider all (or a subset of) possible ways the same information can be encoded.

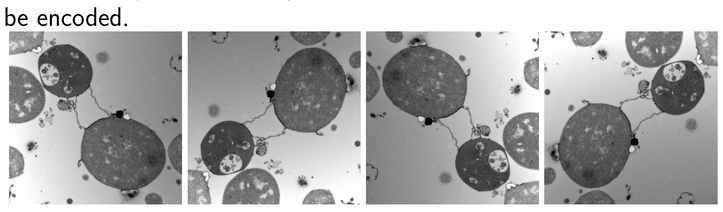

Figure: An example of symmetries applied to a transmission electron micrograph of L-form Bacillus subtilis, as well as a possible way to use data augmentation to cover all possible symmetries in the group of rotations by 90°, 180°, and 270°.

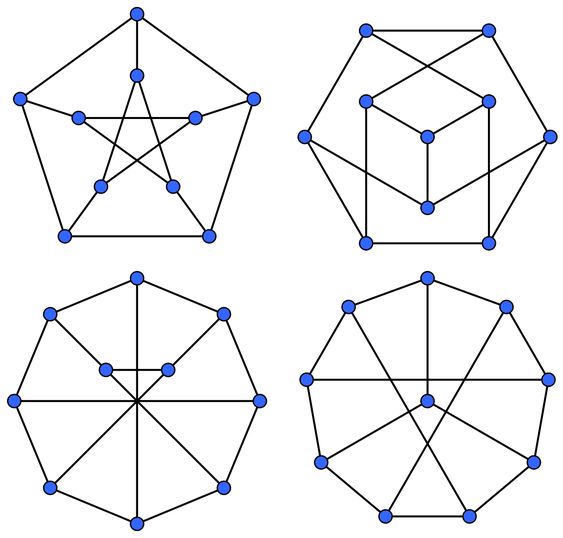

Figure: An example of symmetries in a graph in the form of graph isomorphisms. The four graphs here drawn are the same graph, merely visualized in different ways.

However, data augmentation is computationally expensive and does not impose any constraint on the predictive model itself. Therefore does not guarantee that symmetries of the same datapoint will produce the same result.

Neural Networks

A Feedforward Neural Network computes outputs $\hat{y}$ from an input x through multiple layers of neurons. Each layer applies a linear transformation followed by a nonlinear activation function. Let $W^{(l)}$ be the weight matrix, $b^{(l)}$ the bias vector, and σ a non-linear activation function.
The computation at each layer is:

$$z^{(l)} = W^{(l)}a^{(l)} + b^{(l)}, \quad a^{(l+1)} = \sigma(z^{(l)})$$

where $a^{(0)} = x$. The output of the network after L layers is given by:

$$\hat{y} = \sigma(z^{(L)})$$

During training, the parameters (weight and bias) are updated through an optimization process according to a specific loss function $\mathcal{L}$.

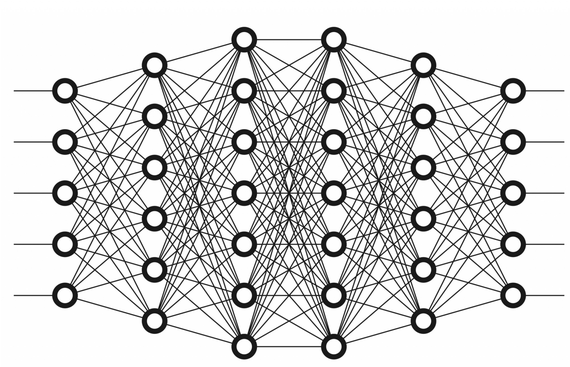

Figure: Visualization of a simple Feedforward Neural Network.

Neural Networks have become in modern times extremely effective at machine learning tasks. However, due to the high number of parameters, **they are difficult to inspect and understand**. Although the mathematics behind Neural Networks is clear, their ability to make predictions is the result of an optimization process on whose final result there is little control on. In particular, they are very susceptible to symmetries: two datapoints which have the same underlying data but differ by symmetry are not guaranteed in any way to produce the same result.

The building blocks of GDL

The building blocks of a GDL Neural Network built upon the symmetry group G are:

- G-equivariant layer
- nonlinearity
- coarsening layer
- G-invariant layer (if required)

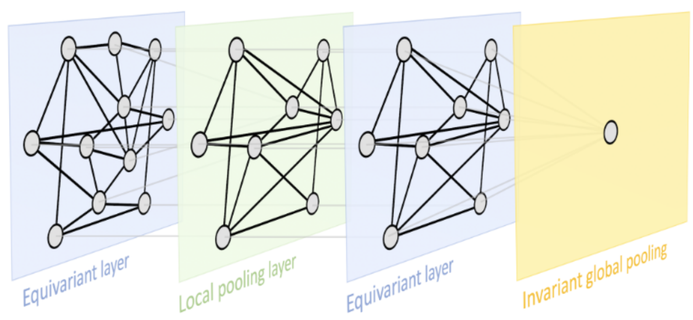

Figure: The GDL blueprint applied to graph data. Courtesy of [1].

Our framework

In our project, we aim to predict **wind speed and direction at a global level** by building a GDL Neural Network based on $SO(3)$ as a symmetry group, i.e. the **group of 3D rotations**. A naive application of either Feedforward or Convolutional networks to a planar projection of the spherical signal is destined to fail. In particular, the space-varying distortions introduced by such a projection will make translational weight sharing in a Convolutional Network ineffective.

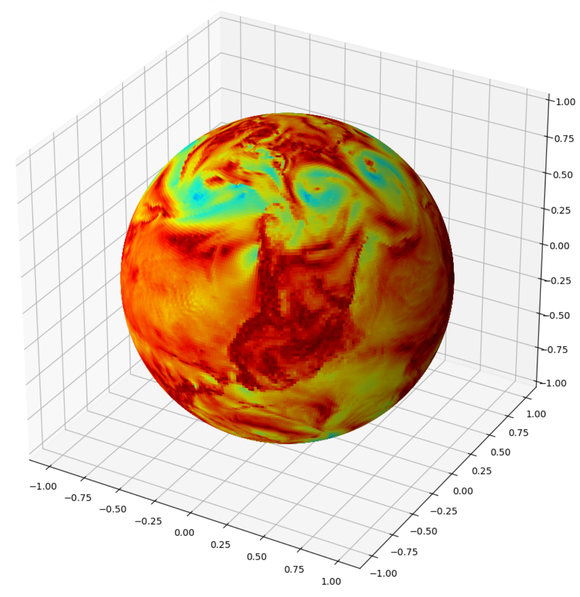

Figure: Wind magnitude plotted on S^2 embedded in $\mathbb{R}^3$. Wind data is given as a vector field in terms of $V(\alpha, \beta)$ and $U(\alpha, \beta)$ the north-bound and east-bound components of the wind at latitude α and longitude β.

The theoretical background can be developed for spheres of any dimension n. Let $M = S^n$ embedded in $\mathbb{R}^{n+1}$. Then we can consider the principal G-bundle

$$\mathrm{SO}(n) \to \mathrm{SO}(n+1) \xrightarrow{\pi} S^n.$$

If we identify $\mathrm{SO}(n)$ as the rotation around the north pole $e_{n+1} = (0, \dots, 1)$, then we can consider $\pi : \mathrm{SO}(n+1) \to S^n$ simply as the map $\pi(A) = Ae_{n+1}$.

Proposition

There is a unique correspondence between real vector fields on the sphere $\xi : S^n \to \mathbb{R}^n$ and equivariant maps $\bar{\xi} : SO(n+1) \to \mathbb{R}^n$ such that $\bar{\xi}(A\hat{B}) = B^{-1}\bar{\xi}(A)$ with

$$\hat{B} = \begin{pmatrix} B & 0 \\ 0 & 1 \end{pmatrix}, \qquad B \in \mathrm{SO}(n), \qquad A \in \mathrm{SO}(n+1).$$

In the specific case of a S^2 embedded in $\mathbb{R}^3$, by using **Euler's angles** we can describe any element of SO(3) in terms of rotations around the Z axis and rotations around the Y axis as

$$Z(\alpha)Y(\beta)Z(\gamma) \in \mathrm{SO}(3)$$

where $\alpha, \gamma \in [0, 2\pi)$ and $\beta \in [0, \pi)$. $L^2(\mathrm{SO}(3), \mathbb{C})$ admits a basis given by *Wigner D-matrices* $D^l := (D^l_{m,n})$ with indices laying in the indexing set

$$I = \left\{ (l, m, n) \ : \ \begin{array}{l} l = 0, 1, 2, 3, \dots, \\ m, n \in [-l, l] \cap \mathbb{Z} \end{array} \right\}.$$

For $(l, m, n) \in I$, define function $D^l_{m,n} : \mathrm{SO}(3) \to \mathbb{C}$ by

$$D^l_{m,n}(Z(\alpha)Y(\beta)Z(\gamma)) = e^{-im\alpha} d^l_{m,n}(\beta) e^{-in\gamma},$$

$$d^l_{m,n}(\beta) = \sqrt{(l+m)!(l-m)!(l+n)!(l-n)!} \cdot \sum_{s=s_0}^{s_1} \frac{(-1)^{m-n+s} \cos\left(\frac{\beta}{2}\right)^{2(l-s)+n-m} \sin\left(\frac{\beta}{2}\right)^{m-n+2s}}{(l+n-s)!s!(m-n+s)!(l-m-s)!}$$

$$f(A) = \sum_{(l,m,n) \in I} \hat{f}^l_{m,n} D^l_{m,n}(A)$$

The G-equivariant layer takes the form of group convolution in the spectral domain:

$$(\Psi * f)(A) = \sum_{l \in \mathbb{N}} \frac{1}{2l+1} \sum_{k,m,n=-l}^{l} \hat{\Psi}^l_{m,k} \hat{f}^l_{k,n} D^l_{m,n}(A)$$

$$\widehat{\Psi * f}^l_{n,m} = \frac{1}{2l+1} \sum_{k=-l}^{l} \hat{\Psi}^l_{m,k} \hat{f}^l_{k,n}$$

Consider the orthogonal decomposition $L^2 = \bigoplus_{n=-\infty}^{\infty} \mathscr{D}_n$, such that for a fixed n then $\mathscr{D}_n$ is spanned by $D^l_{m,n}$, $l \geq n$, $m \in [-l, l] \cap \mathbb{Z}$. We can define an action $\hat{\rho}$ of K on SO(3) by

$$\hat{\rho}(Z(\alpha))A = AZ(-\alpha) = AZ(\alpha)^{-1}$$

and a representation ρ_n on $\mathbb{C}$ by

$$\rho_n(Z(\alpha))z = e^{in\alpha}z.$$

Definition

We say that a function $f \in L^2$ is n-equivariant if

$$f(\hat{\rho}(Z(\alpha))A) = f(AZ(-\alpha)) = e^{in\alpha} f(A) = \rho_n(Z(\alpha)) f(A).$$

Corollary

Vector fields on the sphere, which are in unique correspondence with 1-equivariant functions, are then in unique correspondence with functions spanned by $\mathscr{D}_1$.

We can therefore encode vector fields in a **natively equivariant** fashion by considering functions in $L^2(\mathrm{SO}(3), \mathbb{C})$ spanned by elements of $\mathscr{D}_1$.

For the nonlinearity, the complex analogue to the well-known ReLU can be used:

$$\mathbb{C}\text{-ReLU}(z) = \mathrm{ReLU}(\mathcal{R}(z)) + i\mathrm{ReLU}(\mathcal{I}(z)).$$

References

[1]: Bronstein M., Bruna J., Cohen T., Veličković P., *Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges*, arXiv:2104.13478, (2021).
[2]: Cohen T., Geiger M., Koehler J., Welling M., *Spherical CNNs*, arXiv:1801.10130, (2018).
[3]: Esteves, C., Allen-Blanchette, C., Makadia, A., Daniilidis, K., *Learning SO(3) Equivariant Representations with Spherical CNNs*, Int. J. Comput. Vision, 128(3), 588–600, (2019).

Research supported by the generous contribution of:

TROND MOHN STIFTELSE