Geometric Deep Learning

Author: Juvid Aryaman
Last compiled: February 8, 2022

This document contains my personal notes on Geometric Deep Learning, largely based on Bronstein *et al.* (2021)[1].

1. High-dimensional learning

We discuss the curse of dimensionality in supervised machine learning to motivate why inductive priors are helpful to construct. We'll consider the data domain to be $\mathbb{R}^d$ for this particular discussion.

1.1. Notation

- Data $\mathcal{D} = \{(x_i, y_i)\}_i$, drawn i.i.d. from an underlying data distribution P over $\mathcal{X} \times \mathcal{Y}$.
- Assume data generated by unknown function $y_i = f(x_i)$.
- Assume $\mathcal{X} = \mathbb{R}^d$ and $\mathcal{Y} = \mathbb{R}$.
- The model, or hypothesis class, is a subset $\mathcal{F} \subset \{f : \mathcal{X} \to \mathbb{R}\}$.
- The hypothesis class is assumed to come equipped with a complexity measure of all elements: $\gamma : \mathcal{F} \to \mathbb{R}$. This can usually be defined as a norm, making $\mathcal{F}$ a **Banach space**.
- The (convex) error metric $l(y, y')$, e.g. squared error $l(y, y') = |y - y'|^2$.
- Loss. Consider $\tilde{f} \in \mathcal{F}$
  - Population loss: $\mathcal{R}(\tilde{f}) = \mathbb{E}_P[l(\tilde{f}(x), f(x))]$. This is the true loss of the hypothesis, averaged over the entire data domain.
  - Empirical loss: $\tilde{\mathcal{R}}(\tilde{f}) = 1/n \sum_i l(\tilde{f}(x_i), f(x_i))$. This is the loss over some finite sample $\mathcal{D}$.

1.2. Empirical risk minimization

The underlying goal in supervised learning is to minimise the population loss $\mathcal{R}(\tilde{f})$ given only access to the empirical loss. We seek to construct a bound for the population loss of a hypothesis. Consider $\hat{f} \in \mathcal{F}_\delta$, where $\mathcal{F}_\delta = \{f \in \mathcal{F}; \gamma(f) < \delta\}$, i.e. a hypothesis with bounded complexity. We then decompose $R(\hat{f})$ as follows:

$$\mathcal{R}(\hat{f}) - \inf_{f\in\mathcal{F}} \mathcal{R}(f) = \left(\tilde{\mathcal{R}}(\hat{f}) - \inf_{f\in\mathcal{F}_\delta}\right) + \left[\left(\mathcal{R}(\hat{f}) - \tilde{\mathcal{R}}(\hat{f})\right) - \left(\inf_{f\in\mathcal{F}_\delta} \mathcal{R}(f) - \inf_{f\in\mathcal{F}_\delta} \tilde{\mathcal{R}}(f)\right)\right] + \left(\inf_{f\in\mathcal{F}_\delta} \mathcal{R}(f) - \inf_{f\in\mathcal{F}} \mathcal{R}(f)\right) \tag{1.1}$$

where

- The red term is the population loss of the hypothesis $\hat{f}$ relative to the best-possible hypothesis in the hypothesis class $\mathcal{F}$.
- The blue term is the optimization loss, i.e. how close the empirical loss of the hypothesis gets to the best possible hypothesis in the ball of hypotheses considered $\mathcal{F}_\delta$. Call this ϵ_{opt}.
- The green term is the statistical error, denoting noise from the finite sample used to evaluate the empirical loss, relative to the sampling noise from the best hypothesis in the ball. This can be bounded from above by [**TODO: Didn't understand how**]

$$\epsilon_{\text{stat}} = 2 \sup_{f\in\mathcal{F}_\delta} |\mathcal{R}(f) - \tilde{\mathcal{R}}(f)| \tag{1.2}$$

[1] See also https://geometricdeeplearning.com/

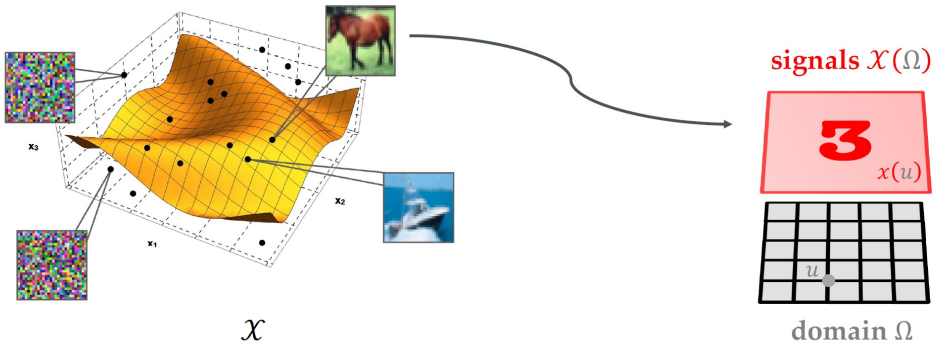

Figure 1. Geometric function spaces will allow us to exploit an underlying low-dimensional structure in the high-dimensional input space $\mathcal{X}$. For example, the space of all possible images are high-dimensional, but "interesting" images will exist on a lower-dimensional manifold embedded in the space of all images. Images off-manifold will look like boring noise. Geometric Deep Learning argues that data will often exist either on a grid, a graph, a group, or a manifold. Each of these have corresponding symmetries which can be leveraged.

- The magenta term is the approximation error, denoting how close the constrained hypothesis class can get to the best function in the unconstrained hypothesis class. Call this ϵ_{approx}.

Thus

$$\mathcal{R}(\hat{f}) \leq \inf_{f \in \mathcal{F}} \mathcal{R}(f) + \epsilon_{\text{opt}} + \epsilon_{\text{stat}} + \epsilon_{\text{approx}}. \tag{1.3}$$

If the hypothesis class is dense then the infimum term is 0, e.g. neural networks with non-polynomial activation (Universal Approximation Theorems). Generally, as the approximation error reduces (through a larger hypothesis space, increasing δ), the statistical error increases.

1.3. Learning Lipschitz functions

Definition 1.1 (Lipschitz function). *A function $f : \mathcal{X} \subseteq \mathbb{R}^d \to \mathbb{R}$ is β-Lipschitz if*

$$|f(x) - f(x')| \leq \beta ||x - x'|| \tag{1.4}$$

i.e. the function cannot vary "too quickly".

If f is 1-Lipschitz, and $P = \mathcal{N}(0, I_d)$, and using empirical risk minimization of the previous section, then a lower-bound on the amount of data required to estimate f up to error ϵ grows as ϵ^{-d}. The curse of dimensionality also crops up in e.g. using a single-layer perceptron to approximate a high-dimensional function, where the statistical error is cursed by dimension.

However, in most cases, data are not simply points in a high-dimensional space: rather, the are **signals** on a low-dimensional **manifold** embedded in a high-dimensional input space $\mathcal{X}$ (Fig. 1). The aim of an inductive prior (which is what Geometric Deep Learning is all about) is to reduce the size of the hypothesis space and thereby reduce statistical error, whilst also keeping the approximation error low, by limiting ourselves to hypothesis spaces which respect the symmetries of the data domain.

2. Geometric priors

2.1. Preliminary definitions

- Assume that data "lives" on a domain Ω. Domain is a **set**, possibly with additional structure
- Data is a **signal** (function) on the domain $x : \Omega \to \mathcal{C}$

- Dimensions of vector space $\mathcal{C}$ are **channels**
- The space of $\mathcal{C}$-valued signals is $\mathcal{X}$, where $\mathcal{X}(\Omega, \mathcal{C}) = \{x : \Omega \to \mathcal{C}\}$

For example, an $n \times n$ RGB image can be considered a function which maps an element of the domain $\Omega = \mathbb{Z}_n \times \mathbb{Z}_n$ onto an element of the vector space $\mathcal{C} = \mathbb{R}^3$.

Definition 2.1 (Addition and scalar multiplication on signals). *Let $x, y \in \mathcal{X}$. Define addition and scalar multiplication of signals through pointwise multiplication over the domain:*

$$(\alpha x + \beta y)(u) = \alpha x(u) + \beta y(u) \tag{2.1}$$

for all $u \in \Omega$ with scalars $\alpha, \beta \in \mathbb{R}$.

Theorem 2.1 (Space of signals is a Hilbert space). *Assuming that $\mathcal{C}$ has an inner product $\langle v, w \rangle_{\mathcal{C}}$, and there exists a measure μ on Ω, we can define an inner product on $\mathcal{X}(\Omega, \mathcal{C})$ as*

$$\langle x, y \rangle = \int_{\Omega} \langle x(u), y(u) \rangle_{\mathcal{C}} \, \mathrm{d}\mu(u). \tag{2.2}$$

Given this inner product, and the fact that Defn. 2.1 implies that $\mathcal{X}$ is a vector space, then the space of signals $\mathcal{X}$ is therefore a Hilbert space.

If Ω is a finite set then μ can be chosen as the counting measure and the integral in Eq.(2.2) becomes a sum. The existence of an inner product allows us to perform "pattern matching", for example comparing a signal x to a filter y. There can be cases where the domain itself are the data: e,g. meshes, or graphs without node or edge features. But we can often turn the domain into a signal on the domain itself, e.g. the adjacency matrix A_{ij} is a signal on $\Omega \times \Omega$.

Note that, in the most general case, data are maps from a point in the domain to a vector space **indexed** by the point in the data domain: $x : \Omega \to \mathcal{C}_u$. For example, the tangent space for a point u on a spherical manifold Ω varies for every point u. In this case, the data aren't functions but **fields** (or **sections of a bundle**), and the space $\mathcal{C}_u$ is called a **fiber**. For simplicity we'll only work with function spaces $\mathcal{X}(\Omega, \mathcal{C})$ for now.

Definition 2.2 (Group). *A* **group**[2] *is a set $\mathfrak{G}$ along with a binary composition operation $\circ : \mathfrak{G} \times \mathfrak{G} \to \mathfrak{G}$ (for brevity, denoted as $\mathfrak{g} \circ \mathfrak{h} = \mathfrak{gh}$) satisfying:*

- *Associativity: $(\mathfrak{gh})\mathfrak{l} = \mathfrak{g}(\mathfrak{hl})$ for all $\mathfrak{g}, \mathfrak{h}, \mathfrak{l} \in \mathfrak{G}$*
- *Identity: there exists a unique $\mathfrak{e} \in \mathfrak{G}$ satisfying $\mathfrak{eg} = \mathfrak{ge} = \mathfrak{g}$ for all $\mathfrak{g} \in \mathfrak{G}$*
- *Inverse: for each $\mathfrak{g} \in \mathfrak{G}$ there exists a unique inverse $\mathfrak{g}^{-1} \in \mathfrak{G}$ such that $\mathfrak{g}\mathfrak{g}^{-1} = \mathfrak{g}^{-1}\mathfrak{g} = \mathfrak{e}$*
- *Closure: The group is closed under composition, i.e. for every $\mathfrak{g}, \mathfrak{h} \in \mathfrak{G}$ we have $\mathfrak{gh} \in \mathfrak{G}$*

Definition 2.3 (Left group action). *If $\mathfrak{G}$ is a group with identity element $\mathfrak{e}$, and X is a set, then a (left) group action α of $\mathfrak{G}$ on X is a function*

$$\alpha : \mathfrak{G} \times X \to X \tag{2.3}$$

that satisfies the following two axioms:

1. *Identity: $\alpha(\mathfrak{e}, x) = x$*
2. *Compatibility: $\alpha(\mathfrak{g}, \alpha(\mathfrak{h}, x)) = \alpha(\mathfrak{gh}, x)$*

for all $\mathfrak{g}$ and $\mathfrak{h}$ in $\mathfrak{G}$ and all x in X. We often shorten $\alpha(\mathfrak{g}, x)$ to $\mathfrak{g}x$ or $\mathfrak{g} \cdot x$ when the action being considered is clear from context.

[2] Disturbingly, the authors choose to use Fraktur font to denote group elements, rather than Lie algebras which is more customary.

Definition 2.4 (Abelian group). *An Abelian group is a particular type of group for which all elements commute, i.e.* $\mathfrak{gh} = \mathfrak{hg}$ *for all* $\mathfrak{g}, \mathfrak{h} \in \mathfrak{G}$.

Importantly, the set of **symmetries** of an object form a group: these are the set of transformations which leave an object invariant.

Definition 2.5 (Generating set of a group). *A generating set of a group is a subset of the group set such that every element of the group can be expressed as a combination of finitely many elements of the subset and their inverses.*

For example, the symmetry group of an equilateral triangle (dihedral group D_3) is generated by a 60° rotation and a reflection. In contrast, the 1D translation group, is generated by infinitesimal displacements – which is an example of a **Lie group** of differentiable symmetries. Note that, in specifying a symmetry group, we need only specify how the group elements **compose**, as opposed to what they *are*. Hence, the same symmetry group can describe very different objects: for example the symmetry group of an equilateral triangle D_3 is the same as the groups of permutations on a sequence of three elements Σ_3.

2.2. Group actions and representations on signals

For a symmetry group $\mathfrak{G}$ operating on the space of signals $\mathcal{X}(\Omega)$, we can express a group action as satisfying

$$(\mathfrak{g} \cdot x)(u) = x(\mathfrak{g}^{-1}u) \tag{2.4}$$

see Fig. 2 for a pictorial depiction of this[3]. Consider x as an image, and $\mathfrak{g}$ to be a finite translation: e.g. a translation to the right. Then $(\mathfrak{g}x)(u)$, i.e. the translated image evaluated at some point u in the domain, is the same as using the original image function, but translating the domain to the left. A consequence of Eq.(2.4) is that group actions on signals is **linear**, in the sense that

$$\mathfrak{g} \cdot (\alpha x + \beta x') = \alpha(\mathfrak{g} \cdot x) + \beta(\mathfrak{g} \cdot x') \tag{2.5}$$

for real scalars α, β and signals $x, x' \in \mathcal{X}$. This linearity means that there exists a **representation** of such groups on signals:

Definition 2.6 (Group representation). *A representation of a group* $\mathfrak{G}$ *on a vector space* V *over a field* K *is a group homomorphism from* $\mathfrak{G}$ *to the general linear group* $GL(V)$

$$\rho : \mathfrak{G} \to GL(V) \tag{2.6}$$

such that

$$\rho(\mathfrak{g}_1\mathfrak{g}_2) = \rho(\mathfrak{g}_1)\rho(\mathfrak{g}_2) \tag{2.7}$$

for all $\mathfrak{g}_1, \mathfrak{g}_2 \in \mathfrak{G}$. *In the case where* V *has a finite dimension* n*, it is common to choose a basis for* V *and identify* $GL(V)$ *with* $GL(n, K)$*, the group of* $n \times n$ *invertible matrices on the field* K.

Note that the particular form of the representation (and how it operates on elements of the vector space) will depend upon the dimensionality of the vector space it is acting on, even when considering the very same underlying group. In practice, even when the vector space Ω is infinite, one must always computationally discretize to find a finite grid, and therefore a finite-dimensional representation exists. I.e. we can find a map $\rho : \mathfrak{G} \to \mathbb{C}^{n \times n}$ that assigns a each group element $\mathfrak{g}$ an invertible matrix $\rho(\mathfrak{g})$. A representation is called **unitary** or **orthogonal** when the matrix $\rho(\mathfrak{g})$ is unitary or orthogonal for all $\mathfrak{g} \in \mathfrak{G}$.

[3] The inverse is required to obtain a valid group action that is associative, satisfying $(\mathfrak{g} \cdot (\mathfrak{h} \cdot x))(u) = ((\mathfrak{gh}) \cdot x)(u)$. [**TODO:** Didn't understand why this wouldn't work without the inverse]

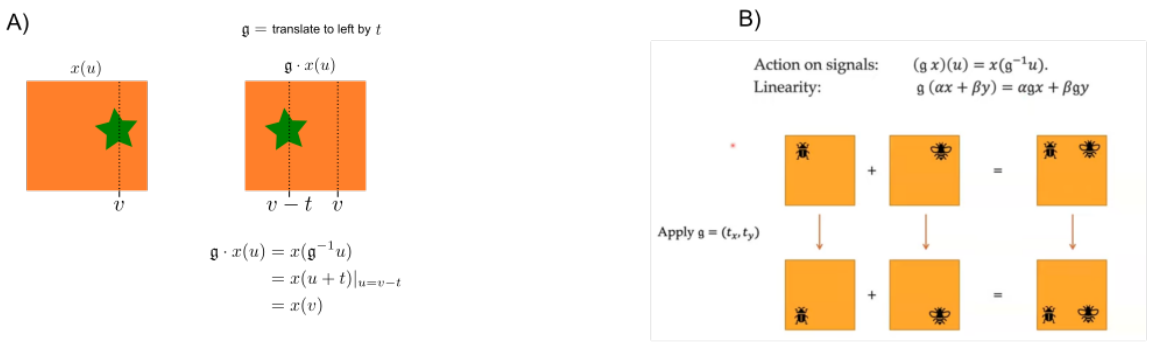

Figure 2. Group action is linear on the space of signals. A) Consider a binary image x, a point on the underlying 2D grid u, and a symmetry group element such as a translation of the image to the left by t pixels. Eq.(2.4) shows the effect of the group action on the image, which is to act inversely on the image domain: i.e. the image grid slides to the right by t pixels and the same image is placed at that location. B) Linearity on signals means that if we add two signals, and then apply a symmetry transformation, that is the same as applying the symmetry transformation on each signal separately and then adding the result together.

2.3. Invariant and equivariant functions

The symmetry of the domain Ω underlying signals $\mathcal{X}(\Omega)$ imposes structure on the function mapping the signal to the target $\mathcal{Y}$. Two important cases are invariant and equivariant functions.

Definition 2.7 (Invariant function). *A function $f : \mathcal{X}(\Omega) \to \mathcal{Y}$ is $\mathfrak{G}$-invariant if*

$$f(\rho(\mathfrak{g})x) = f(x) \tag{2.8}$$

for all $\mathfrak{g} \in \mathfrak{G}$ and $x \in \mathcal{X}(\Omega)$. I.e. its output is unaffected by the group action on the input.

An example of an invariant function might be in image classification, where an image contains a particular object (e.g. a dog or a cat), and the abstract function f mapping the image onto the true label is invariant to the position of the object in the image. In such settings, we often implement a convolutional neural network as the model. However, CNNs are not shift-invariant, but shift-equivariant.

Definition 2.8 (Equivariant function). *A function $f : \mathcal{X}(\Omega, \mathcal{C}) \to \mathcal{X}(\Omega', \mathcal{C}')$ is $\mathfrak{G}$-equivariant if*

$$f(\rho(\mathfrak{g})x) = \rho'(\mathfrak{g})f(x) \tag{2.9}$$

for all $\mathfrak{g} \in \mathfrak{G}$. I.e. group action on the input affects the output in the same way, albeit with the possibility of the input and output spaces having different domains.

A prototypical example of shift-equivariance is image segmentation: a segment mask must follow shifts in the input image. In deep learning, it is often not a good idea to build an invariant representation "too soon". In order to recognise an object, a model must first recognize its parts – this is why neural networks should be deep. If representations of parts are invariant, then information about the *relative pose* of parts is lost, see Fig. 3A. Hence we want equivariant, not invariant, layers for the early parts of the network.

2.4. Isomorphisms and automorphisms

Definition 2.9 (Isomorphism). *An isomorphism is a structure-preserving mapping $\eta : \Omega \to \Omega'$ between two structures Ω and Ω' that can be reversed by an inverse mapping. For example, if $\Omega = \{0, 1, 2\}$ and $\Omega' = \{a, b, c\}$, then the bijection $\eta(0) = a$, $\eta(1) = b$, $\eta(2) = c$ is an set isomorphism.*

Definition 2.10 (Automorphism). *An automorphism is an isomorphism $\tau : \Omega \to \Omega$, mapping a mathematical object onto itself. For example, if $\Omega = \{0, 1, 2\}$ then the cyclic shift $\tau(u) = u + 1$ mod 3 is an automorphism.*

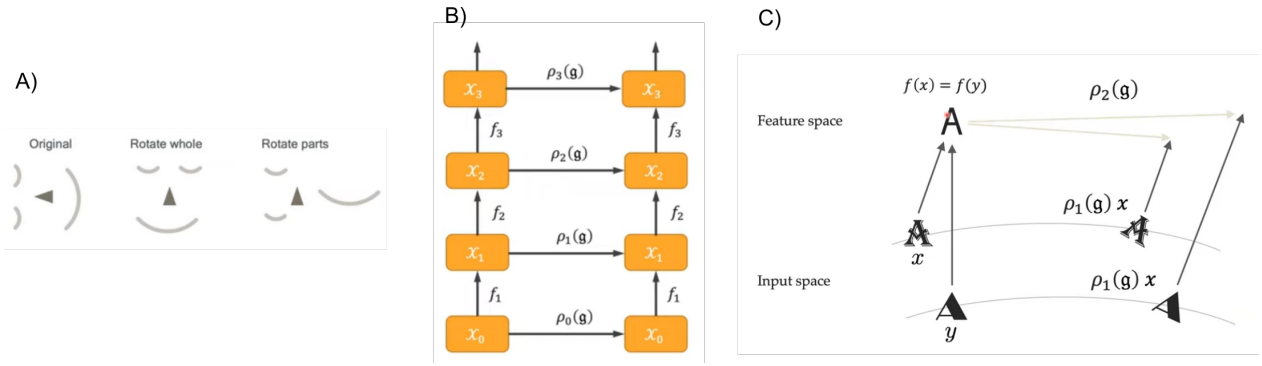

Figure 3. Equivariant layers are necessary in the early part of a neural network. A) In order for a neural network to recognise a face, it must first build representations of each of the parts: eyes, ears, mouth, nose. If the entire face is rotation invariant, that does not mean that each of the parts should be represented as rotation invariant, otherwise the relative orientation between the parts can be lost. B) On the left we have a sequential neural network, with layers f_i and corresponding input signals $\mathcal{X}_{i-1}$. The arrows from left-to-right denote a symmetry transformation on the feature space. The image says that applying the symmetry transformation on the input (e.g. $\mathcal{X}_0$) and then applying the layer (e.g. f_1), should be the same as applying the layer and then applying the same symmetry transformation on the output(albeit on a different representation, e.g. $\mathcal{X}_1$). In other words, each layer obeys Eq.(2.9). I.e. $f_i \circ \rho_{i-1}(\mathfrak{g}) = \rho_i(\mathfrak{g}) \circ f_i$ for each layer i. C) In practice, many neural network architectures will not generalize consistently across **orbits** of input. In this example of a CNN, rotating the input image will not result in a rotated representation but rather send the input space to different places in the feature space, because CNNs are not generally rotation equivariant.

2.5. Scale separation prior

A scale separation prior makes the assumption that it is possible to separate interactions across scales: that local interactions propagate upwards to higher scales, rather than small scales directly interacting with large scales. A multiscale coarsening of a domain Ω into a hierarchy $\Omega_1, ..., \Omega_J$ requires us to also define a hierarchy of signals $\mathcal{X}_j(\Omega_j, \mathcal{C}_j) := \{x_j : \Omega_j \to \mathcal{C}_j\}$.

Definition 2.11 (Locally stable function (informal)). *A function $f : \mathcal{X}(\Omega) \to \mathcal{Y}$ is locally stable at scale j if it admits a factorization of the form*

$$f \approx f_j \circ P_j \tag{2.10}$$

where $P_j : \mathcal{X}(\Omega) \to \mathcal{X}(\Omega_j)$ is a non-linear **coarse graining** *and $f_j : \mathcal{X}(\Omega_j) \to \mathcal{Y}$.*

The reason for why coarse graining is useful is because **local decompositions are stable with respect to small deformations of the domain**. Bronstein *et al.* (2021) discuss by analogy how decomposing signals in terms of **wavelets** (a local decomposition) instead of Fourier transforms (a global decomposition) renders the representation stable when we only have *approximate* invariance to e.g. translations (see Fig. 5).

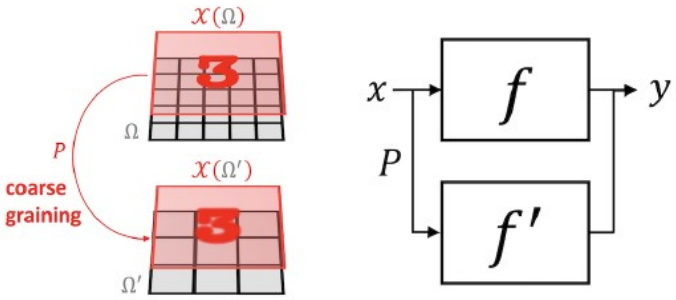

Figure 4. Illustration of scale separation for image classification. The classifier f' defined on signals on the coarse grid $\mathcal{X}(\Omega')$ should satisfy Eq.(2.10).

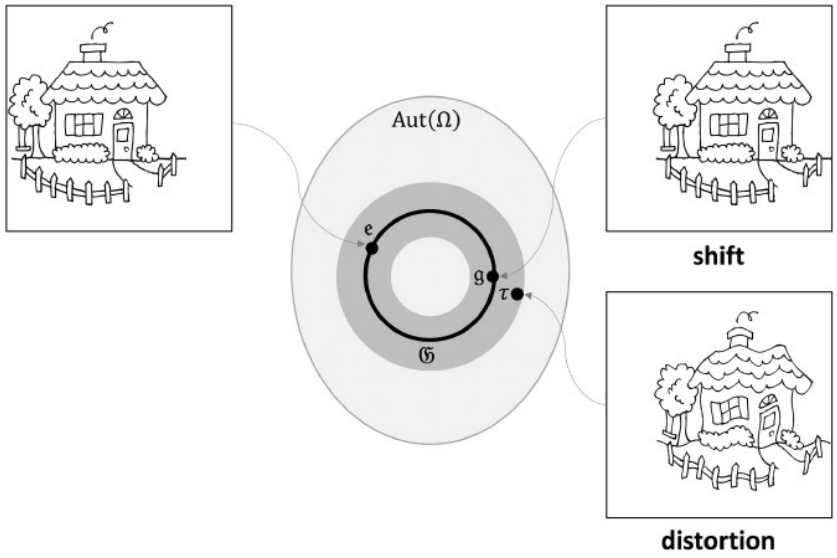

Figure 5. Defining "approximate" invariance as transformations of the domain "near" to a group orbit. The set of all bijective mappings from Ω to itself is the set automorphism group $\mathrm{Aut}(\Omega)$. A symmetry group, such as the translation group, is a subgroup of $\mathrm{Aut}(\Omega)$. It is possible to define a distance metric between a group element of $\mathrm{Aut}(\Omega)$ and a given symmetry group. It is possible to define an approximate invariance group $\mathfrak{G}$ (gray ring) by admitting group elements $\tau \in \mathrm{Aut}(\Omega)$ within some tolerance of the pure symmetries (black circle). In this case, the ring represents the set of shifts with small distortions.

References

Bronstein, M. M., J. Bruna, T. Cohen, and P. Veličković, 2021 Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. arXiv preprint arXiv:2104.13478 .