Geometric Deep Learning

Author: Juvid Aryaman
Last compiled: January 5, 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/

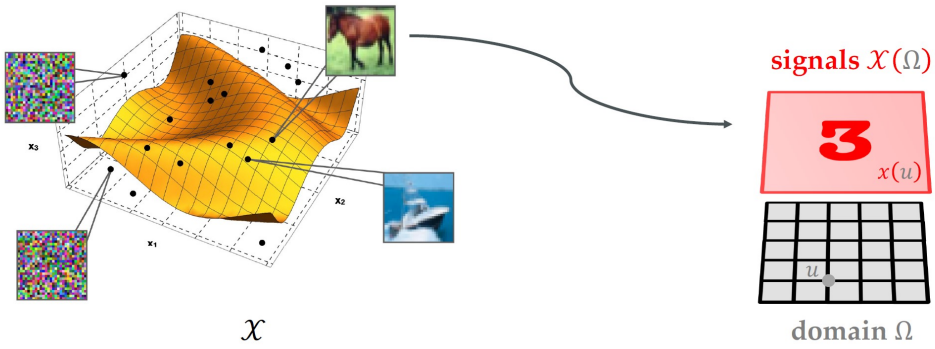

Figure 1. Geometric function spaces will allow us to exploit an underlying low-dimensional structure in the high-dimensional input space $\mathcal{X}$. For example, the space of all possible images are high-dimensional, but "interesting" images will exist on a lower-dimensional manifold embedded in the 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. Notation

- 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.

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 .