ON CHARACTERIZING THE CAPACITY OF NEURAL NETWORKS WITH ALGEBRAIC TOPOLOGY

COLTON GRAINGER

- inspired by my colleague Nikki Sanderson's work with Professors Meiss and Bradley
- she just defended her PhD, titled *Topological Data Analyses of Time Series Using Witness Complexes*

1. Introduction

- In this talk, we apply TDA to the question "is depth needed for deep learning?"
  - as most of you probably know, the word "deep" has become very very popular in machine learning over the last few years, due to the dramatic resurgence of multi-layer neural networks
  - an informal argument for why they're so good, atleast in terms of their expressiveness, is that they can compactly represent very complex nonlinear predictors, for instance in the context of vision, one might have a multilayer network that computes relatively simple features, like edges, then on deeper nodes more sophisticated features, like faces and cats
  - well, this is just a hand-waving argument: can we formalize it? in some sense, we already have.
  - since Cybenko in 1989 we know that any continuous function on a bounded domain in R^{d} can be accurately approximated by a 2-layer network that's sufficiently large
  - the catch is that to get this approximation, we have to use networks that are exponentially large in the dimension d
  - the model selection problem is posed as so: given a limited number of hidden units, how can we determine the optimal width and depth of a network?
- Guss and Salakhutdinov's main contribution to answer is empirical
- For each architecture in a catalog of neural nets of varying width and depth, Guss and Salakhutdinov trained 100 instances on 930 synthetic data sets of varying topological complexity, reporting out the minimum and average error versus the number of minibatches seen during training.
- Guss conjectures that the complexity of a dataset strictly limits the *expressivity* of a neural architecture.
- **The goal of this talk is to make this characterization more precise.**

1.1. depth vs width.

- Guss presents evidence that the capacity of a feed-forward neural network to accurately approximate the positive decision region of point-cloud a dataset depends on:
  - the *depth*, or number of hidden layers in the architecture
  - the *width*, or number of hidden units in the architecture
  - and a summary statistic of the dataset called its *persistent homology*

Date: 2018-11-27.
Compiled: 2018-11-27.

1.2. **what's a feed-forward neural network?**

- composed of single neurons:

$$\mathbf{x} \mapsto \sigma(\langle \mathbf{w}, \mathbf{x} \rangle + b), \qquad \mathbf{x}, \mathbf{w} \in \mathbf{R}^d, \quad b \in \mathbf{R}.$$

  - computes some linear transformation of the vector
  - then pushes it through some nonlinearity,
    - could be a sigmoid, could be a ReLu: $z \mapsto \max\{0, z\}$.
  - they're arranged in layers: we'll assume a single output neuron

- depth 2 width n network:

$$\mathbf{x} \mapsto \sum_{i=1}^{n} v_i \sigma(\langle \mathbf{w}_i, \mathbf{x} \rangle + b_i)$$

- depth 3 width n network:

$$\mathbf{x} \mapsto \sum_{i=1}^{n} u_i \sigma \left(\sum_{j=1}^{n} v_i \sigma(\langle \mathbf{w}_{i,j}, \mathbf{x} \rangle + b_{i,j}) + c_i \right)$$

2. Problem: Model Selection

2.1. **exhaustive search.** For each $\ell \in \{1, \ldots, 6\}$ and $h_0 \in \{1, \ldots, 500\}$, (we assume deeper layers are of fixed width, i.e., $h_1, \ldots, h_\ell = \beta_0(\mathcal{D})$), take

- fully connected architectures of (ℓ, h_i) layers and hidden units in the first hidden layer,
  - with 2 input neurons (coordinates of a point in the unit square I^2)
  - with unit weights initialized to samples from a normal distribution $\mathcal{N}(0, 1/\beta_0(\mathcal{D}))$,
  - and rectified linear (ReLu) activation functions.

3. Key Idea: Consider Homological Complexity

- we need to define a finite set of topological invariants called Betti numbers
- the historical champion here is the *homology* functor, whose definition I will loosely sketch
- now, Guss and Salakhutdinov (among many others) applied *persistent homology* as a scale-free method for computing a *barcode* of Betti numbers from a point cloud dataset
  - it's like an imperfect fourier transform
  - the point cloud is like the signal in the time-domain,
  - the topological *barcode* is like the decomposition in the frequency domain

The main steps in a persistent homology analysis are as follows.

- We treat each data point as a node in a graph, drawing edges between nearby nodes where nearby is according to a scale parameter.
- We form complexes from the simplices formed by the nodes and edges, and examine the topology of the complexes as a function of the scale parameter.
- The topological features such as connected components, and holes of various dimensions that persist across scales are the ones that capture the underlying shape of the dataset.

In greater detail:

3.1. **data.** We make a minimal interpretation of the point cloud.

We'll only need single linkage clustering.

- no model, no probability
- just pairwise distances
- familiar idea: points in a metric space

3.2. **filtered cell complexes.** From the point cloud, we build a filtered cell complex (think about the construction parameterized by a time t).

The easiest version:

- 0 and 1 cells are just vertices and edges
- at some time near zero, we've constructed a weighed graph where the edges are labelled with pairwise distances between points

What happens in higher order interactions? We add higher dimensional connections, a little more abstract, sufficient for computation:

Say at some time, 3 vertices can be fit within a ball of radius 5. Then we add a 2-cell, a triangle, filling in the edges of the 1-cell skeleton.

- 2-cells are triangles
- 3-cells are tetrahedra (they connect 4 data points).

Then it's tricky to visualize—enter homological algebra.

3.3. **homological algebra.** Like with graphs, where one has adjacency matrices recording how edges and vertices fit together, so also, each cell complex has an associated matrix recording how:

- triangles touch the 3 edges in their boundary
- tetrahedra touch the 4 triangles in their boundary
- and so on

So, for each complex in our filtration, we obtain a matrix. At the 0-1 level, these matrices are exactly adjacency matrices, but then they quickly become more abstract. We're cataloging how cells of higher dimension fit together with cells of one lower dimension

At some point in time (in the filtration) we'll know everything we need to about the relations

0. edges to vertices
1. triangles to edges
2. tetrahedra to triangles
3. and so on

What does one do when they have a matrix?

- compute column and null spaces
- these correspond to boundary and cycle groups
- now the boundary group of dimension $n+1$ is a subgroup of the cycle group of dimension n,
- the quotient group is called the n^{th} homology group: it's an abelian group
- the rank of each dimension's homology group produce sthe invariants we desire: Betti numbers for each dimension n, measuring the number of generators for n-cycles

- namely, their conjecture that there's a lower bound on the number of hidden units in the first layer required to accurately approximate a positive decision region

5. References

- "The Power of Depth for Feedforward Neural Networks - YouTube"[1]. Retrieved November 27, 2018.
- "Dr Vidit Nanda, University of Oxford - YouTube"[2]. Retrieved November 27, 2018.

[1] https://www.youtube.com/watch?v=Ue_hR6x0B-U
[2] https://www.youtube.com/watch?v=JqajfI4-WnM