Jacob Miller

Training with Topology

Using the Euler Characteristic Transform

December 13, 2024

Outline

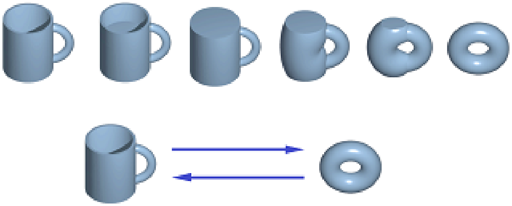
What is topology?

What is topology?

Topology is the study of shape up to deformations. In the mind of a topologist, a coffee cup is the same as a donut.

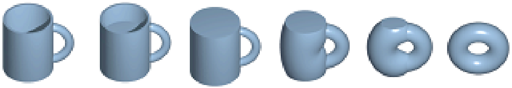
What is topology?

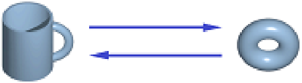
Topology is the study of shape up to deformations. In the mind of a topologist, a coffee cup is the same as a donut.



What is topology?

Topology is the study of shape up to deformations. In the mind of a topologist, a coffee cup is the same as a donut.





Remember that, as topologists, we cannot:

1. cut
2. glue

Topological Data Analysis:

GOAL:

Topological Data Analysis:

GOAL: Convince you that thinking about data as samples from a high-dimensional manifold is correct.

Topological Data Analysis:

GOAL: Convince you that thinking about data as samples from a high-dimensional manifold is correct.

- We will use topological methods to train on the MNIST dataset.

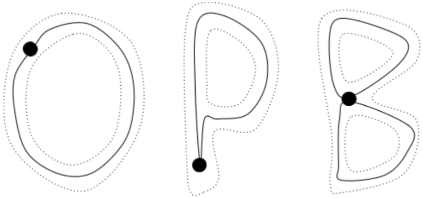
Glaring Issue: Topology cannot distinguish some numbers/letters!

Topological Data Analysis:

GOAL: Convince you that thinking about data as samples from a high-dimensional manifold is correct.

- We will use topological methods to train on the MNIST dataset.

Glaring Issue: Topology cannot distinguish some numbers/letters!



- Ignore this issue for now...

Simplices: Building Blocks

Simplicial complexes are a common way to describe shapes in topology

Definition (Simplex)

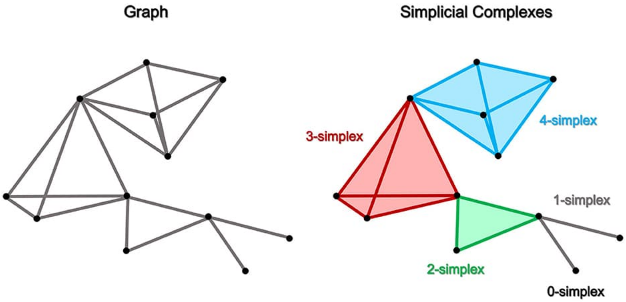
A *k-simplex* is the convex hull of $k+1$ affinely independent points:

Simplices: Building Blocks

Simplicial complexes are a common way to describe shapes in topology

Definition (Simplex)

A *k-simplex* is the convex hull of $k+1$ affinely independent points:



Simplicial Complexes

Definition

A simplicial complex Δ is a collection of simplices satisfying:

Simplicial Complexes

Definition

A simplicial complex Δ is a collection of simplices satisfying:

1. If $\sigma \in \Delta$ is a simplex, then Δ contains all faces of σ
2. If two simplices in Δ intersect, their intersection is a face of each

Example:

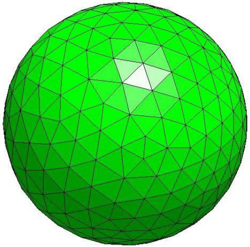
Simplicial Complexes

Definition

A simplicial complex Δ is a collection of simplices satisfying:

1. If $\sigma \in \Delta$ is a simplex, then Δ contains all faces of σ
2. If two simplices in Δ intersect, their intersection is a face of each

Example:



Euler Characteristic for Simplicial Complexes

Definition

The Euler characteristic $\chi(\Delta)$ of a finite simplicial complex Δ is:

$$\chi(\Delta) = \sum_{i=0}^{\infty} (-1)^i f_i$$

where f_i is the number of i-dimensional simplices in Δ.

Euler Characteristic for Simplicial Complexes

Definition

The Euler characteristic $\chi(\Delta)$ of a finite simplicial complex Δ is:

$$\chi(\Delta) = \sum_{i=0}^{\infty} (-1)^i f_i$$

where f_i is the number of i-dimensional simplices in Δ.

Example

A filled in triangle (a.k.a. 2-simplex):

- $f_0 = 3$ (vertices), $f_1 = 3$ (edges), $f_2 = 1$ (faces)
- $\chi(\text{triangle}) = 3 - 3 + 1 = 1$

The Mapper Algorithm

Working under the assumption that our data cloud was sampled from a manifold M:

The Mapper Algorithm

Working under the assumption that our data cloud was sampled from a manifold M:

Input: A data cloud D in an ambient space $\mathbb{R}^n$.

The Mapper Algorithm

Working under the assumption that our data cloud was sampled from a manifold M:

Input: A data cloud D in an ambient space $\mathbb{R}^n$.

Ingredients:

- A lens (filter) function $f : \mathbb{R}^n \to \mathbb{R}$
- An open cover for $\mathbb{R}$
- A clustering algorithm (We used Affinity Propagation)

The Mapper Algorithm

Working under the assumption that our data cloud was sampled from a manifold M:

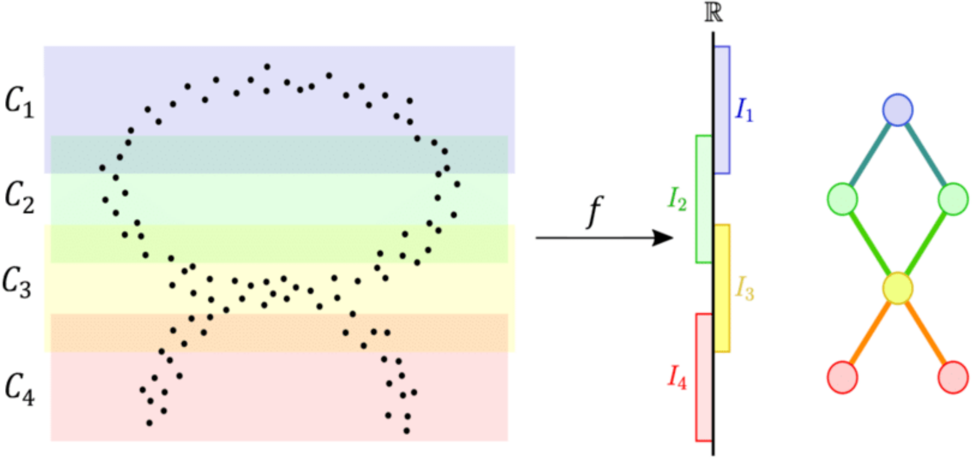
Input: A data cloud D in an ambient space $\mathbb{R}^n$.

Ingredients:

- A lens (filter) function $f : \mathbb{R}^n \to \mathbb{R}$
- An open cover for $\mathbb{R}$
- A clustering algorithm (We used Affinity Propagation)

Output: A simplicial complex – a topological approximation of the manifold M.

Diagram of Mapper



Euler Characteristic Transform

Input: An embedded simplicial complex Δ – normalized to live in a unit ball in $\mathbb{R}^n$.

Euler Characteristic Transform

Input: An embedded simplicial complex Δ – normalized to live in a unit ball in $\mathbb{R}^n$.
Output: A function

$$ECT : S^{n-1} \times [-1, 1] \to \mathbb{Z}$$

ECT Setup

Given an embedded simplicial complex Δ and a direction $\vec{w} \in S^{n-1} \subseteq \mathbb{R}^n$ we can define a function $F_{\vec{w}} : \Delta \to \mathbb{R}$ inductively:

$$F_{\vec{w}}(v) := v \cdot \vec{w} \qquad \text{For any vertex } v \in \Delta$$

ECT Setup

Given an embedded simplicial complex Δ and a direction $\vec{w} \in S^{n-1} \subseteq \mathbb{R}^n$ we can define a function $F_{\vec{w}} : \Delta \to \mathbb{R}$ inductively:

$$F_{\vec{w}}(v) := v \cdot \vec{w} \qquad \text{For any vertex } v \in \Delta$$

If $\sigma = [v_1, \ldots, v_k] \in \Delta$ then

$$F_{\vec{w}}(\sigma) := \max \{ F_{\vec{w}}(v_i) \mid 1 \leq i \leq k \}$$

ECT Setup

Given an embedded simplicial complex Δ and a direction $\vec{w} \in S^{n-1} \subseteq \mathbb{R}^n$ we can define a function $F_{\vec{w}} : \Delta \to \mathbb{R}$ inductively:

$$F_{\vec{w}}(v) := v \cdot \vec{w} \qquad \text{For any vertex } v \in \Delta$$

If $\sigma = [v_1, \ldots, v_k] \in \Delta$ then

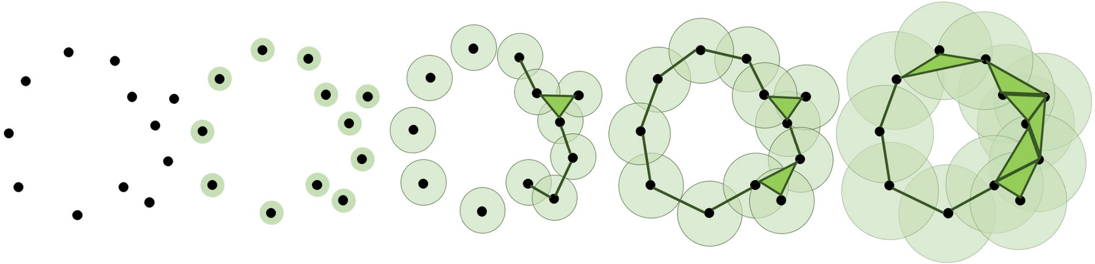
$$F_{\vec{w}}(\sigma) := \max \{ F_{\vec{w}}(v_i) \,|\, 1 \leq i \leq k \}$$

For any threshold value $t \in [-1, 1]$

$$F_{\vec{w}}^{-1}[[-1, t)] \text{ is a subcomplex of } \Delta$$

An example of a filtration

Warning: This is a picture of a Vietoris-Rips filtration



K_1 K_2 K_3 K_4 K_5

Recap

If we pick a direction and a threshold value, we get a simplicial complex.

Recap

If we pick a direction and a threshold value, we get a simplicial complex.

Changing our threshold value will give us different subcomplexes of Δ.

Recap

If we pick a direction and a threshold value, we get a simplicial complex.

Changing our threshold value will give us different subcomplexes of Δ.

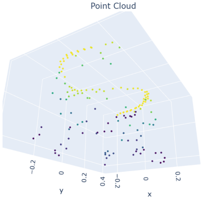
IDEA: For each direction and threshold value, calculate the Euler characteristic!

ECT Definition

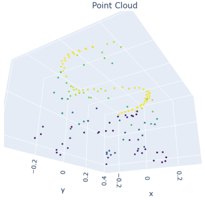
The Euler Characteristic Transform ECT is given by

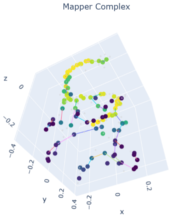
$$ECT(\vec{w}, t) := \chi\left(F_{\vec{w}}^{-1}[(-\infty, t)]\right)$$

ECT Results

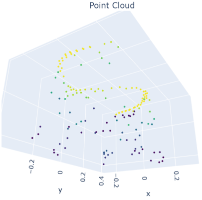


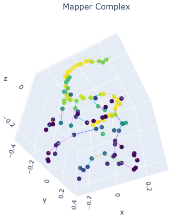
ECT Results

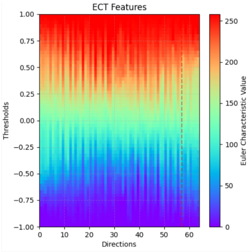

Point Cloud
−0.2
0
0.2
0.4
y
−0.2
0
0.2
x


Mapper Complex
z
0
−0.2
−0.4
−0.2
0
0.2
0.4
y
−0.2
0
0.2
x

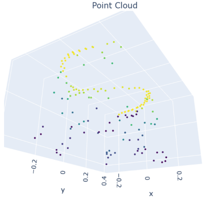
ECT Results

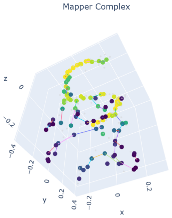

Point Cloud
−0.2
0
0.2
0.4
y
−0.2
0
0.2
x

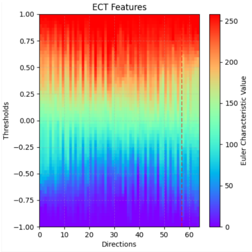

Mapper Complex
z
0
−0.2
−0.4
−0.2
0
0.2
0.4
y
−0.2
0
0.2
x


ECT Features
Thresholds
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
10
20
30
40
50
60
Directions
Euler Characteristic Value
250
200
150
100
50
0

ECT Results





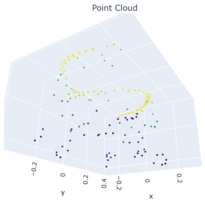


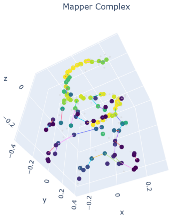
Theorem (Turner et al., 2014)

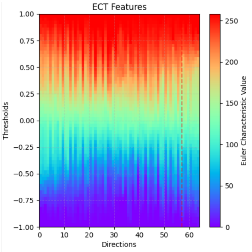
The ECT is ***injective*** *on the space of constructible* sets in* $\mathbb{R}^d$. *So:*

$$\text{If } ECT(K) \neq ECT(K'), \text{ then } K \neq K'.$$

ECT Results







Theorem (Turner et al., 2014)

*The ECT is **injective** on the space of constructible* sets in $\mathbb{R}^d$. So:*

$$\textit{If } ECT(K) \neq ECT(K'), \textit{ then } K \neq K'.$$

So the above theorem is true for simplicial complexes in $\mathbb{R}^d$.

Preprocessing

- Load image from MNIST dataset
- Filter out empty pixels
- Normalize pixel and grayscale values to live in the unit 3-ball
- Translate by the mean to standardize
- Apply some Gaussian smoothing (to look good)

Processing

- Load data cloud
- Compute mapper complex using zen-mapper
  - filter function: Projection to the line $y + x = 0$.
  - Covering Scheme: Width Balanced with 10 elements, 40% overlap
  - Clusterer: Affinity Propagation with 0.9 damping (random preferences)
- Embed into $\mathbb{R}^3$
- Approximate ECT
  - 64 directions (sampled via Fibonacci spiral)
  - 64 thresholds (uniformly distributed in $[-1, 1]$)

ECTNet Architecture

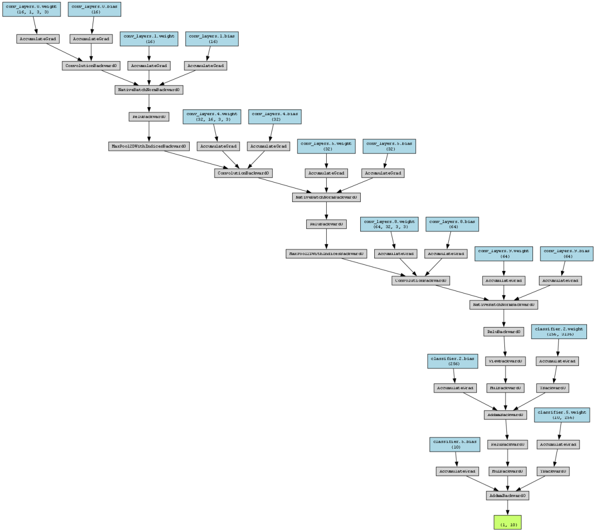
Designing your own CNN - Sanjay Dutta
Creating a CNN from Scratch using Pytorch - Abhishek

- **Convolutional Layers:**
  - Conv1: 1 $\rightarrow$ 16 channels (3$\times$3)
  - BatchNorm + ReLU + MaxPool(2$\times$2)
  - Conv2: 16 $\rightarrow$ 32 channels (3$\times$3)
  - BatchNorm + ReLU + MaxPool(2$\times$2)
  - Conv3: 32 $\rightarrow$ 64 channels (3$\times$3)
  - BatchNorm + ReLU
- **Classifier Head (Make Decision):**
  - Flatten
  - Dropout (0.3) – randomly deactivates neurons
  - Linear: input_dim $\rightarrow$ 256
  - ReLU + Dropout(0.3)
  - Linear: 256 $\rightarrow$ 10

Simplified:

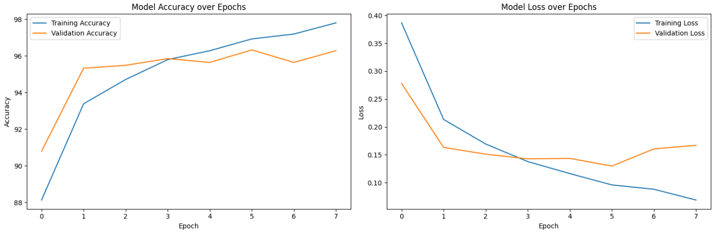
- Two main components:
  - Feature extraction (conv layers)
  - Classification (dense layers)
- Convolutional section:
  - Progressively expands features
  - Spatial reduction via max pooling (takes maximum value from the input region)
- Classifier section:
  - Flattens features
  - Final 10-class prediction

Torchviz Diagram:

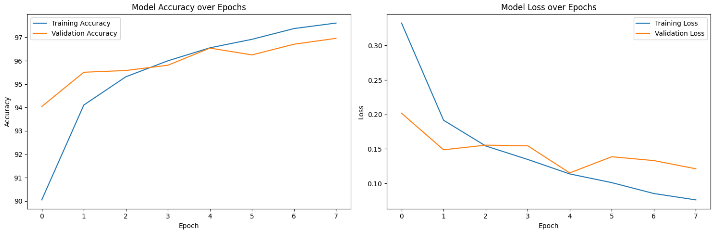


Training Results:

Mapper Complex Training Results: 97.8% accuracy



Point Complex Training Results: 97.6% accuracy



IOWA

Thank you!

Thanks for listening! If you want to see the implementation checkout the GitHub:

https://github.com/Jamiller137/ect-mnist

There is an interactive app that will let you draw a number and see what the model is doing.

IOWA