Topological Analysis of Decision Boundaries

Maxim Mikhaylov with supervision by Dr. Patrick Schnider

ETH Zürich

Sep 2024 – Feb 2025

Profile

Engineering

- **Core:** computational methods, including numerical implementation (C++)
- **Skills:**
  - Systems programming
  - Optimization
  - Parallel programming
  - Numerical methods

I build fast, reliable software for computationally intensive research tasks.

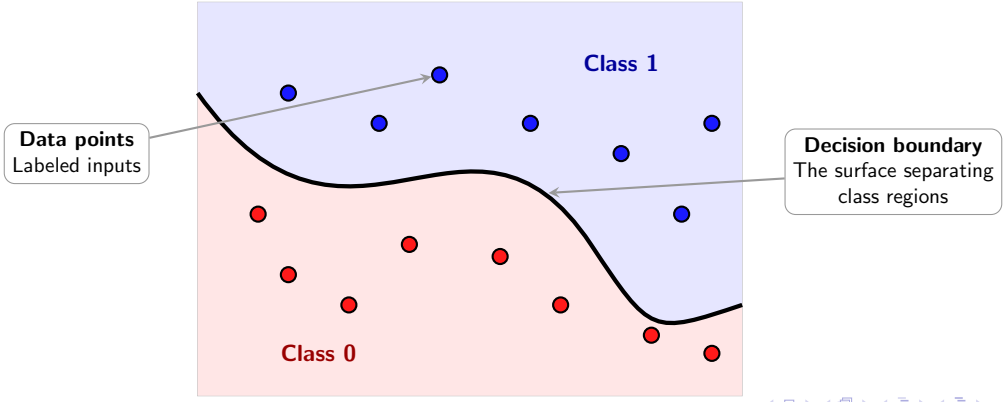
Research

- **Focus:** physical systems and interdisciplinary problems
- **Experience:**
  - Robotics (ITMO)
  - Satellite Nav (Bosch)
  - Topological Data Analysis (ETH)

I research physical reality with computational methods.

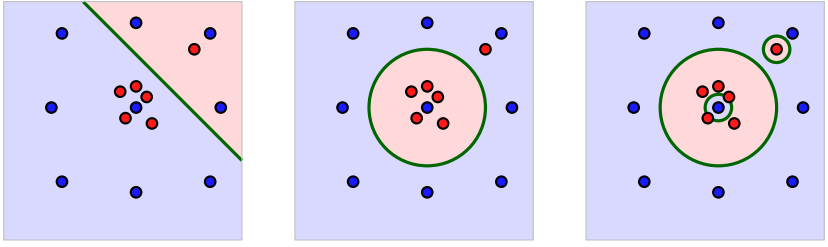
Project Introduction

- Neural networks partition high-dimensional input spaces into class regions.
- The boundaries between these regions, *decision boundaries*, are important for generalization.
- Hypothesis: The topological complexity (e.g., fragmentation, holes) of these surfaces correlates with overfitting.



Theoretical background: Overfitting

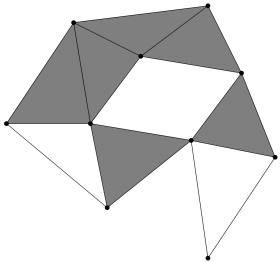
- Model fits training data too closely, fails to generalize to unseen data
- Performance improves on training data, but degrades on validation data
- Maybe this is reflected in the topology of decision boundaries?

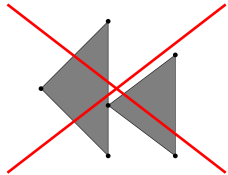


Underfit Fit Overfit

Theoretical background: Persistent homology

- Simplicial complex: a collection of simplices glued together "nicely"





Theoretical background: Persistent homology

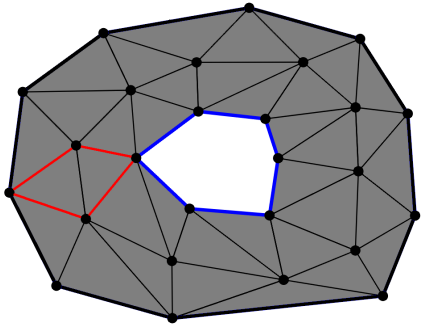
- Simplicial complex: a collection of simplices glued together "nicely"
- Homology groups of dimension p: measure the number of p-dimensional holes in a simplicial complex

Theoretical background: Persistent homology

- Simplicial complex: a collection of simplices glued together "nicely"
- Homology groups of dimension p: measure the number of p-dimensional holes in a simplicial complex
  - $p = 0$: connected components

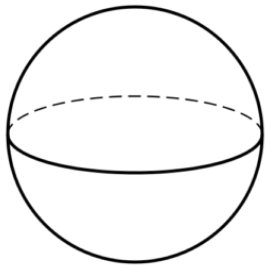
Theoretical background: Persistent homology

- Simplicial complex: a collection of simplices glued together "nicely"
- Homology groups of dimension p: measure the number of p-dimensional holes in a simplicial complex
  - $p = 0$: connected components
  - $p = 1$: unfilled cycles



Theoretical background: Persistent homology

- Simplicial complex: a collection of simplices glued together "nicely"
- Homology groups of dimension p: measure the number of p-dimensional holes in a simplicial complex
  - $p = 0$: connected components
  - $p = 1$: unfilled cycles
  - $p = 2$: unfilled voids

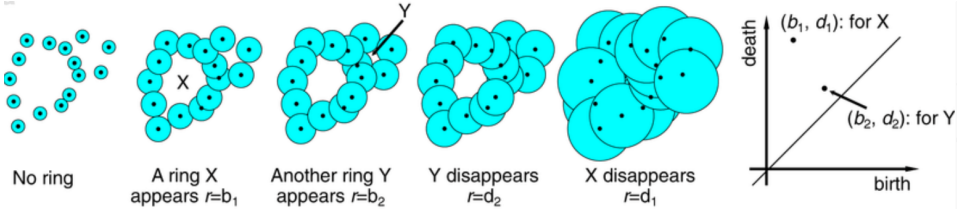


Theoretical background: Persistent homology

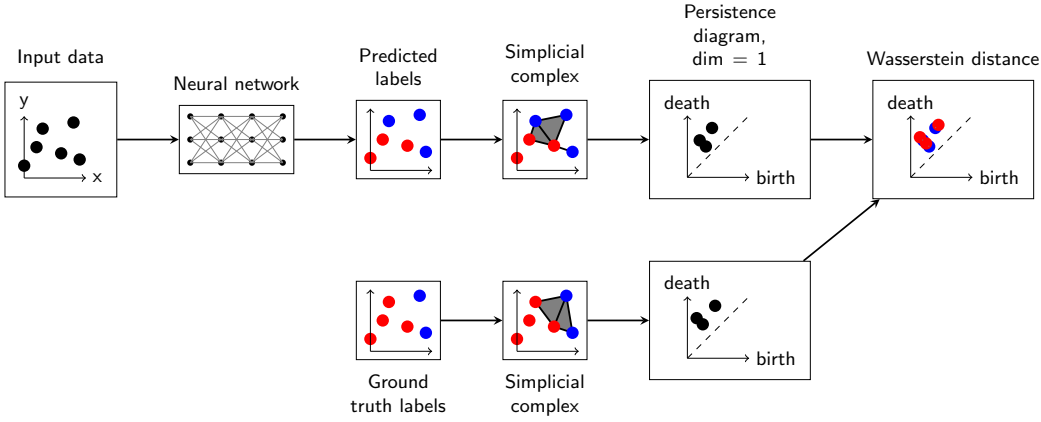
- Simplicial complex: a collection of simplices glued together "nicely"
- Homology groups of dimension p: measure the number of p-dimensional holes in a simplicial complex
  - $p = 0$: connected components
  - $p = 1$: unfilled cycles
  - $p = 2$: unfilled voids
- *Persistent* homology: homology groups of a simplicial complex as it evolves

Theoretical background: Persistent homology

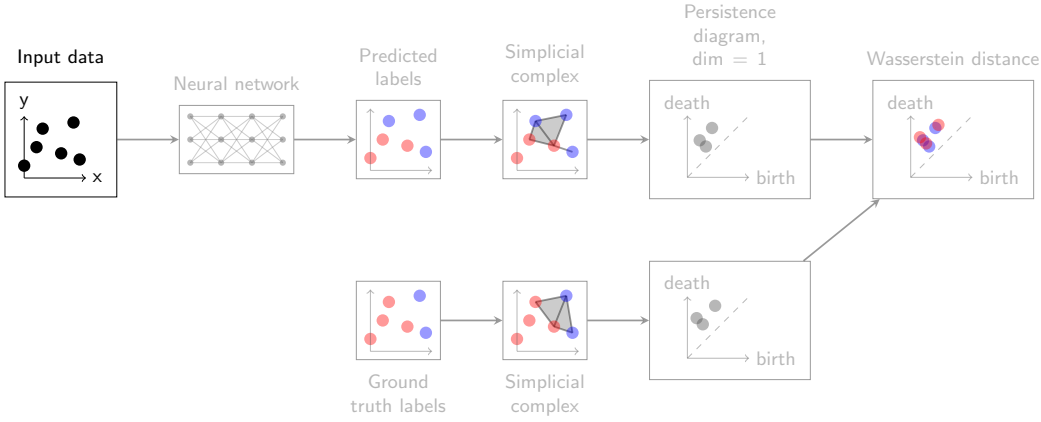
- Simplicial complex: a collection of simplices glued together "nicely"
- Homology groups of dimension p: measure the number of p-dimensional holes in a simplicial complex
  - $p = 0$: connected components
  - $p = 1$: unfilled cycles
  - $p = 2$: unfilled voids
- *Persistent* homology: homology groups of a simplicial complex as it evolves
- Example: growing balls



Pipeline to capture decision boundary topology



Pipeline

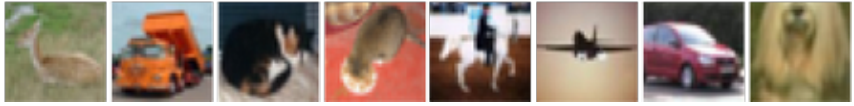

Input data
y
x
Neural network
Predicted labels
Simplicial complex
Persistence diagram, dim = 1
death
birth
Wasserstein distance
death
birth
Ground truth labels
Simplicial complex
death
birth

Input data

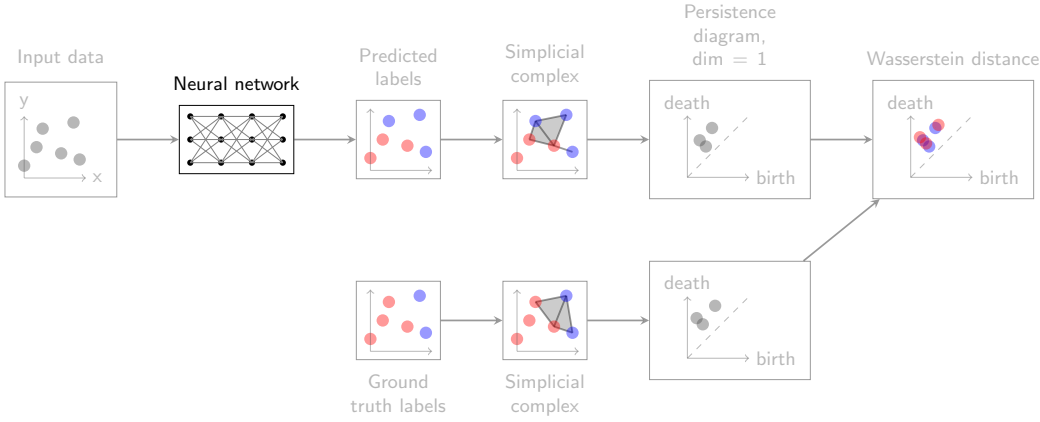
- MNIST: 28x28 grayscale images of handwritten digits, dim = 784

- FashionMNIST: 28x28 grayscale images of fashion articles, dim = 784

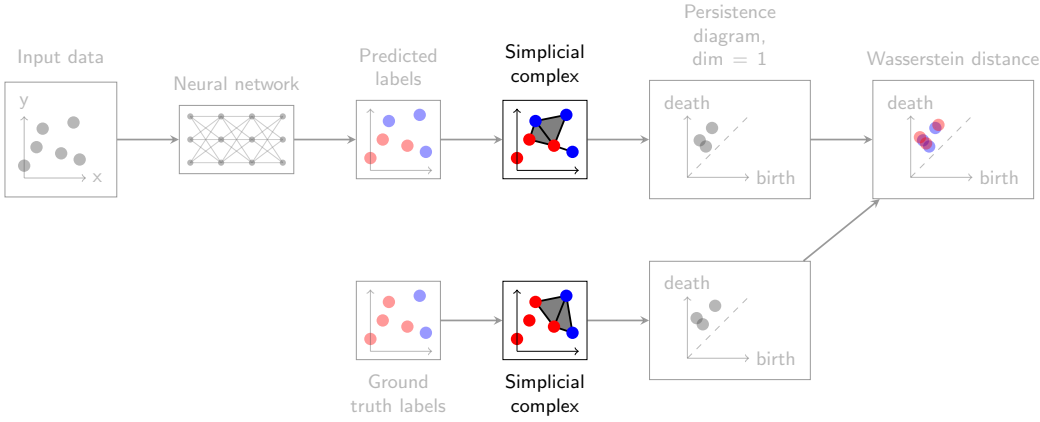
- CIFAR-10: 32x32 color images of 10 classes, dim = 3072



Pipeline


Input data
y
x
Neural network
Predicted labels
Simplicial complex
Persistence diagram, dim = 1
death
birth
Wasserstein distance
death
birth
Ground truth labels
Simplicial complex
death
birth

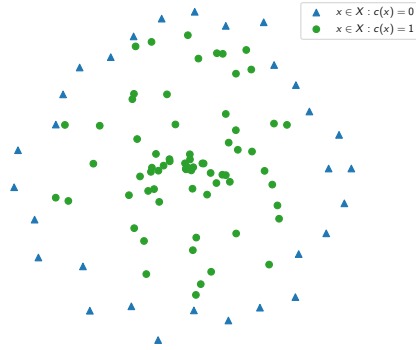
Pipeline


Input data
y
x
Neural network
Predicted labels
Simplicial complex
Persistence diagram, dim = 1
death
birth
Wasserstein distance
death
birth
Ground truth labels
Simplicial complex
death
birth

Labeled Vietoris-Rips complex

- Set of points X
- Labels $c : X \to \mathbb{Z}_k$
- Parameter ε

Constructed in three steps:

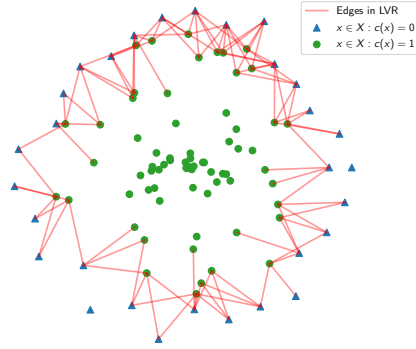


Labeled Vietoris-Rips complex

- Set of points X
- Labels $c : X \to \mathbb{Z}_k$
- Parameter ε

Constructed in three steps:

1. Create a graph G_ε with vertex set X by adding an edge between points $x_i, x_j \in X$ iff:
   - $\|x_i - x_j\| \leq \varepsilon$ (points are close enough)
   - $c(x_i) \neq c(x_j)$ (different classes)

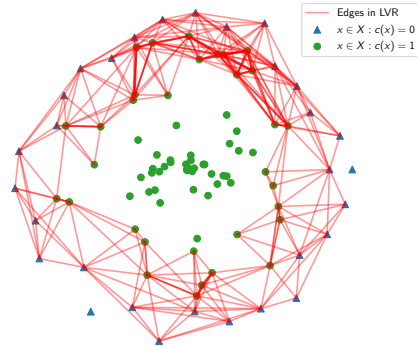


Labeled Vietoris-Rips complex

- Set of points X
- Labels $c : X \to \mathbb{Z}_k$
- Parameter ε

Constructed in three steps:

1. Create a graph G_ε with vertex set X by adding an edge between points $x_i, x_j \in X$ iff:
   - $\|x_i - x_j\| \leq \varepsilon$ (points are close enough)
   - $c(x_i) \neq c(x_j)$ (different classes)
2. Add edges between all 2-hop neighbors in G_ε

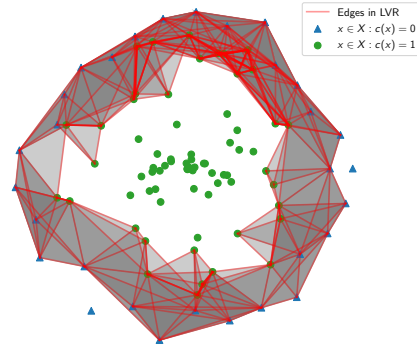


Labeled Vietoris-Rips complex

- Set of points X
- Labels $c : X \to \mathbb{Z}_k$
- Parameter ε

Constructed in three steps:

1. Create a graph G_ε with vertex set X by adding an edge between points $x_i, x_j \in X$ iff:
   - $\|x_i - x_j\| \leq \varepsilon$ (points are close enough)
   - $c(x_i) \neq c(x_j)$ (different classes)
2. Add edges between all 2-hop neighbors in G_ε
3. Standard Vietoris-Rips construction: include simplex if all faces are included



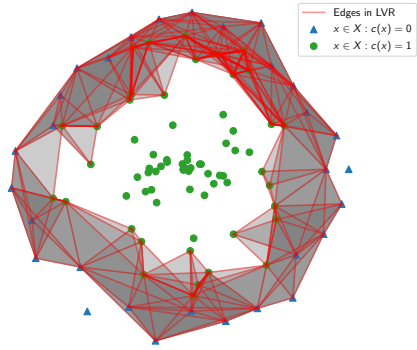
Labeled Vietoris-Rips complex

- Set of points X
- Labels $c : X \to \mathbb{Z}_k$
- Parameter ε

Constructed in three steps:

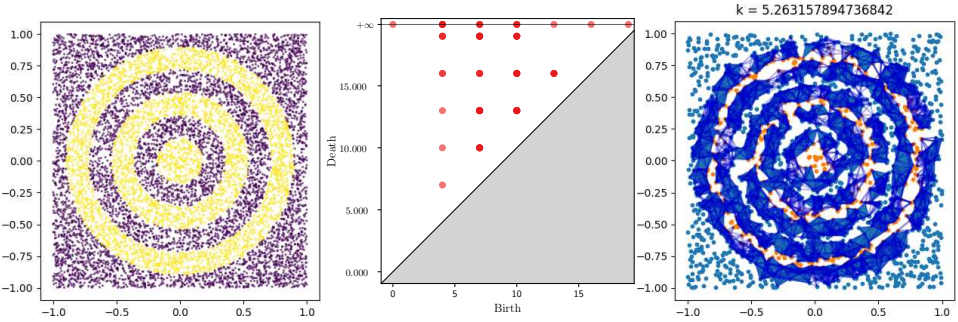
1. Create a graph G_ε with vertex set X by adding an edge between points $x_i, x_j \in X$ iff:
   - $\|x_i - x_j\| \leq \varepsilon$ (points are close enough)
   - $c(x_i) \neq c(x_j)$ (different classes)
2. Add edges between all 2-hop neighbors in G_ε
3. Standard Vietoris-Rips construction: include simplex if all faces are included

- Simplices cross the boundary
- Applicable to multiple classes



Synthetic 2D experiemnts

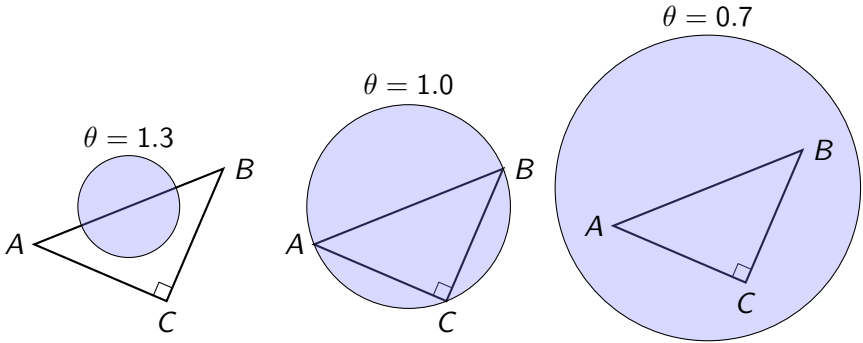
To see if the LVR complex recovers the homology of the decision boundary, we use synthetic 2D data with nested annuli.



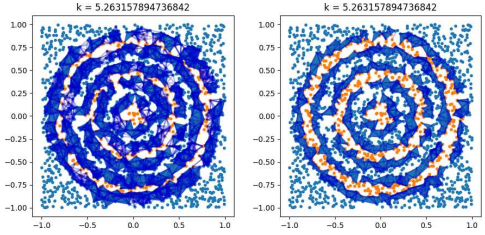
Edges cross the decision boundary multiple times $\implies$ spurious holes appear.

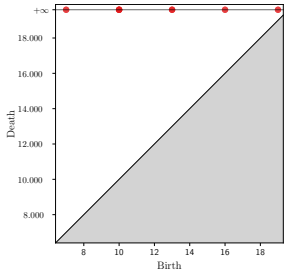
Circumcircle filtering

- Remove an edge AB if exists vertex $C : |AB|^2 > (|AC|^2 + |BC|^2)\theta$.
- θ is a parameter to relax the condition.



2D Validation





Computational Bottleneck:

- Requires checking the condition for every edge against every other node.
- Python prototypes were prohibitively slow.

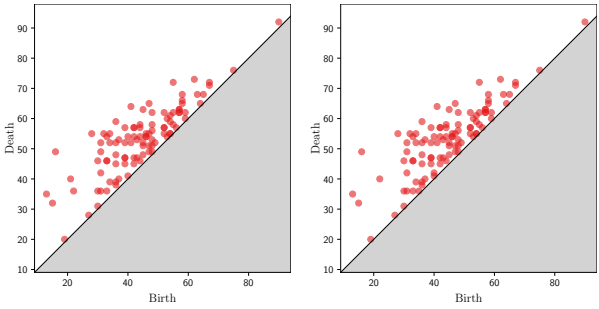
C++ Implementation:

- C++ SIMD extension to handle the filtering.
- **Outcome:** Reduced filtering time to negligible overhead compared to complex construction.
- Enabled scaling the analysis to large, high-dimensional datasets.

Circumcircle filtering impact on high-dimensional data

Dataset	Binary	Multiclass
MNIST	7	4
FashionMNIST	14	1
CIFAR10	7	0

Maximum differences in Wasserstein distance with and without CC.

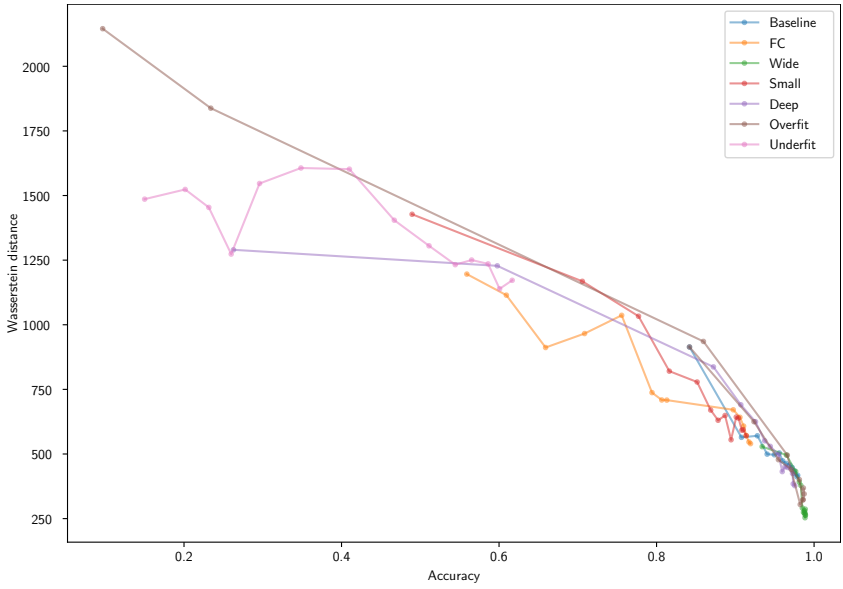


The two persistence diagrams that differ the most when CC is applied.

The C++ optimization allowed us to rigorously test the projects hypothesis.
Conclusion: In the studied high-dimensional cases, sparsity renders filtering unnecessary, validating the use of the standard LVR complex for these tasks.

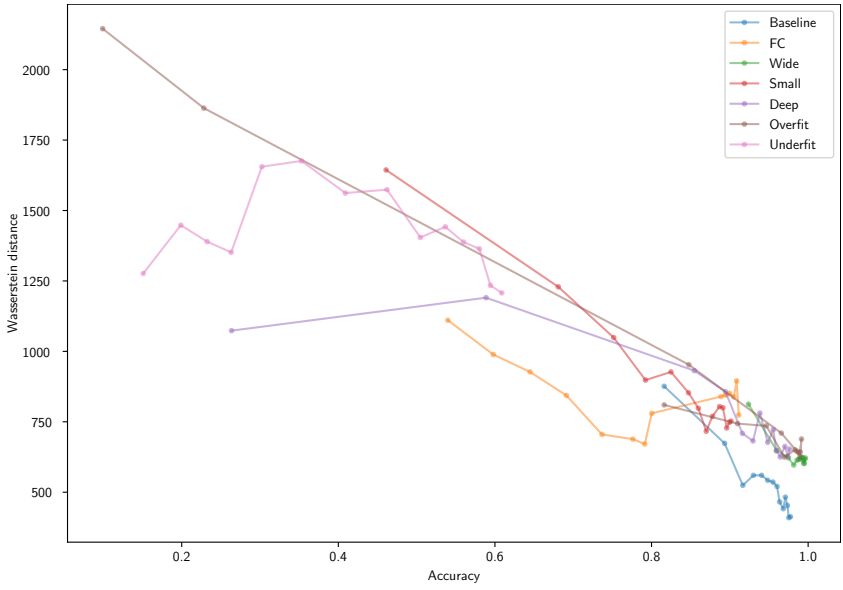
Results: multiclass classification

Clear indication of overfit in the overfit model's trajectory



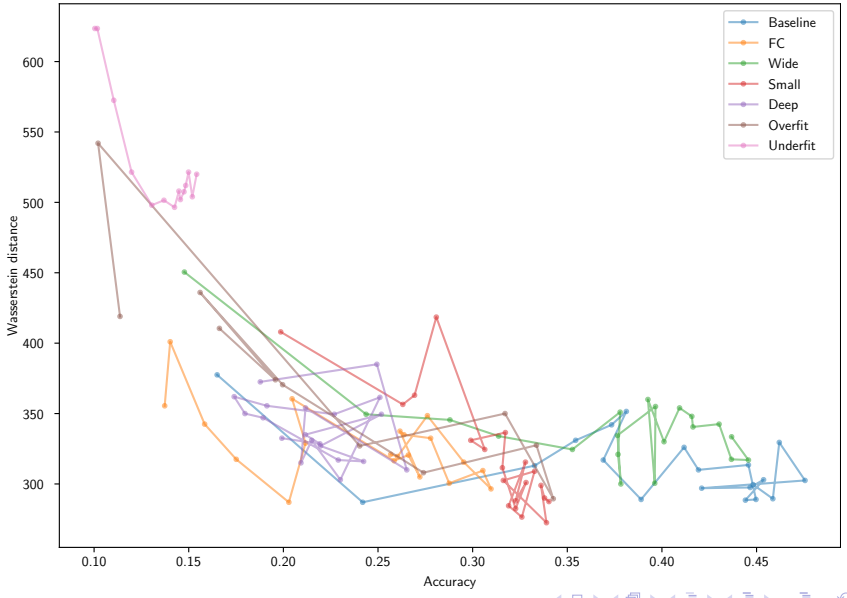
Results: multiclass classification, train data

TDA on training data, accuracy on test data



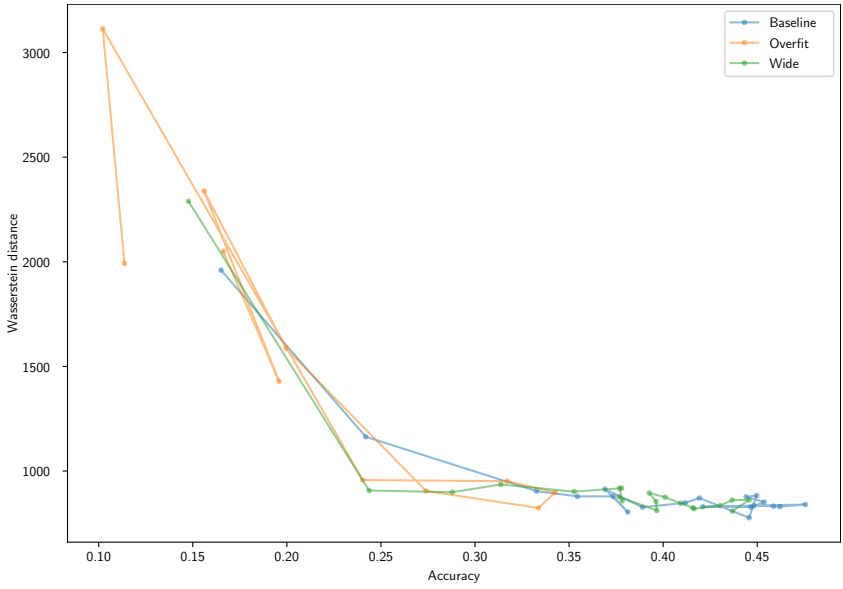
Results: multiclass classification, CIFAR10, 2000 points

Weak correlation as dimensionality of data increases, number of datapoints is unchanged



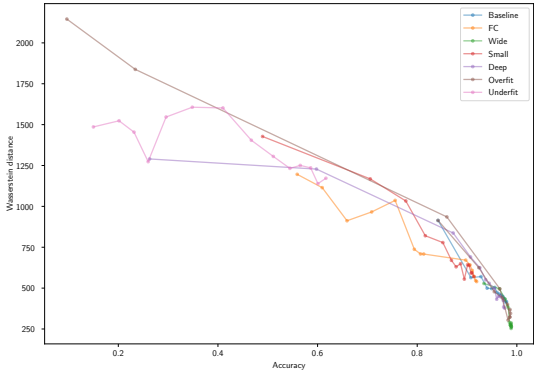
Results: multiclass classification, CIFAR10, 8000 points

Increasing number of points improves correlation



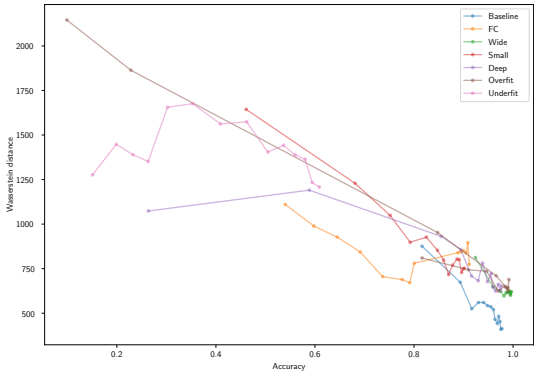
Conclusion

- TDA provides insights into classifier behavior
  - Strong correlation between Wasserstein distance and model accuracy
  - Wasserstein distance increases during overfitting



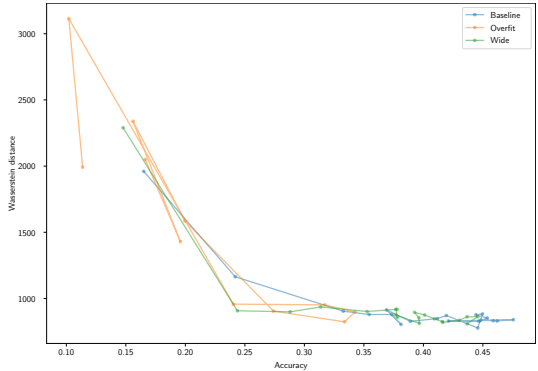
Conclusion

- TDA provides insights into classifier behavior
- Training data decision boundary topology correlates with test performance
  - A-priori indicator for model behavior
  - Enables model evaluation without separate test sets
  - Could serve as early overfitting detection



Conclusion

- TDA provides insights into classifier behavior
- Training data decision boundary topology correlates with test performance
- Dimensionality scaling from CIFAR-10 experiments
  - Initial effectiveness decrease in higher dimensions
  - Performance restored by increasing sample size
  - Limitation: too many points required for very high-dimensional X



Research Identity

My research interest lies consistently at the intersection of **computational systems** and **physical reality**.

Robotics (ITMO)

- *Context:* Autonomous agents with strict compute limits.
- *Challenge:* ML inference under extreme resource constraints.

Satellite Nav (Bosch)

- *Context:* Global positioning systems with noisy signal data.
- *Challenge:* Robust processing of noisy and partially observable signals.

Topology (ETHZ)

- *Context:* High-dimensional geometric data analysis.
- *Challenge:* Highly-efficient C++ kernels for high-dimensional geometric predicates.

Common Thread: Understanding how domain-specific constraints shape computational models.

Motivation: Why SimuCell3D?

The Challenge:

- Biology is the ultimate "physical system"—massive scale, complex geometry, and noise.
- *SimuCell3D* provides a strong framework for tissue simulation, but exploring new biological regimes at tissue scale poses modelling challenges.

I want to apply my background in research and C++ engineering to:

1. **Extend Capabilities:** Implement the Reaction-Diffusion solvers (PDEs) required for chemical concentration awareness.
2. **Scale Performance:** Optimize the core mechanics (collision/solvers) to run larger simulations faster.
3. **Scientific Enablement:** Expand the range of biological questions that can be addressed through simulation.

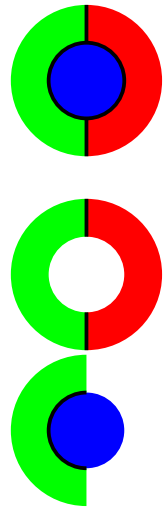
Extra slides

Binarization

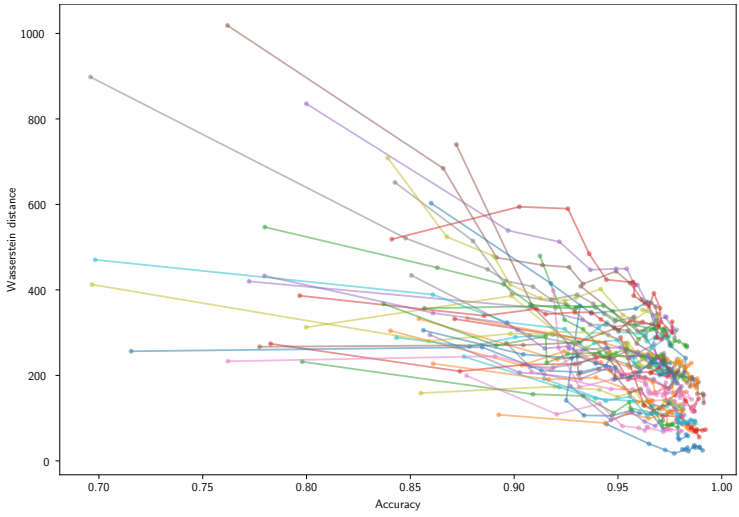
Previous work only considered binary classification by splitting a dataset with n classes into $\binom{n}{2}$ binary datasets. This changes the homology groups:

- Original has $H_1 \ncong 0$
- All binary decompositions have $H_1 \cong 0$

It is better to avoid binarization.



Results: binary classification



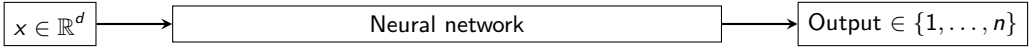
Relationship between Wasserstein distance and model accuracy across all binary classification pairs in MNIST. Connected points represent consecutive epochs for the same class pair.

Theoretical background: Machine learning

- Learn from data without explicit programming

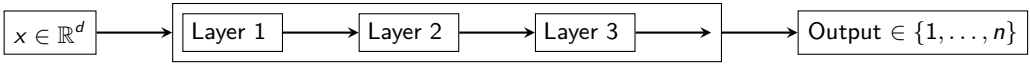
Theoretical background: Machine learning

- Learn from data without explicit programming
- Classifiers map input data $\in \mathbb{R}^d$ to class labels $\in \{1, \ldots, n\}$



Theoretical background: Machine learning

- Learn from data without explicit programming
- Classifiers map input data $\in \mathbb{R}^d$ to class labels $\in \{1, \ldots, n\}$
- Networks are a sequence of layers, data flowing from input layer to output layer

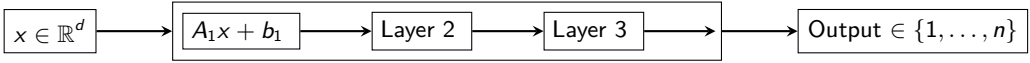


Theoretical background: Machine learning

- Learn from data without explicit programming
- Classifiers map input data $\in \mathbb{R}^d$ to class labels $\in \{1, \ldots, n\}$
- Networks are a sequence of layers, data flowing from input layer to output layer
- To train, compute the gradient of the loss function with respect to the network parameters using backpropagation, and update parameters using gradient descent

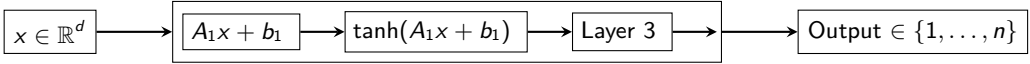
Theoretical background: Machine learning

- Learn from data without explicit programming
- Classifiers map input data $\in \mathbb{R}^d$ to class labels $\in \{1, \ldots, n\}$
- Networks are a sequence of layers, data flowing from input layer to output layer
- To train, compute the gradient of the loss function with respect to the network parameters using backpropagation, and update parameters using gradient descent
- Fully-connected layers: $y = Ax + b$



Theoretical background: Machine learning

- Learn from data without explicit programming
- Classifiers map input data $\in \mathbb{R}^d$ to class labels $\in \{1, \ldots, n\}$
- Networks are a sequence of layers, data flowing from input layer to output layer
- To train, compute the gradient of the loss function with respect to the network parameters using backpropagation, and update parameters using gradient descent
- Fully-connected layers: $y = Ax + b$
- Activation layers: nonlinearity for expressiveness



Theoretical background: Machine learning

- Learn from data without explicit programming
- Classifiers map input data $\in \mathbb{R}^d$ to class labels $\in \{1, \ldots, n\}$
- Networks are a sequence of layers, data flowing from input layer to output layer
- To train, compute the gradient of the loss function with respect to the network parameters using backpropagation, and update parameters using gradient descent
- Convolutional layers: parameter-efficient filters for images / time series