Topological Regularizer

Tao Hou, CS410/510

University of Oregon

Acknowledgment

Contents of the slide (including figures) are based on the paper:

- Chen, Chao, Xiuyan Ni, Qinxun Bai, and Yusu Wang. "A topological regularizer for classifiers via persistent homology." In The 22nd International Conference on Artificial Intelligence and Statistics, pp. 2573-2582. PMLR, 2019.

Overview

- Regularization plays a crucial role in supervised learning: a successfully regularized model strikes a balance between a *perfect description of the training data* and *the ability to generalize to unseen data*

Overview

- Regularization plays a crucial role in supervised learning: a successfully regularized model strikes a balance between a *perfect description of the training data* and *the ability to generalize to unseen data*
- The authors propose to enforce the structural simplicity of the classification boundary by regularizing over its *topological complexity*
- The measurement of topological complexity incorporates the importance of topological features (e.g., connected components, handles, and so on), and provides a direct control over spurious topological structures

Overview

- Regularization plays a crucial role in supervised learning: a successfully regularized model strikes a balance between a *perfect description of the training data* and *the ability to generalize to unseen data*
- The authors propose to enforce the structural simplicity of the classification boundary by regularizing over its *topological complexity*
- The measurement of topological complexity incorporates the importance of topological features (e.g., connected components, handles, and so on), and provides a direct control over spurious topological structures
- They incorporate the new measurement as a topological penalty in training classifiers
- They also propose an algorithm to compute the *partial derivatives* (gradient) of such penalty

Overview

- Regularization plays a crucial role in supervised learning: a successfully regularized model strikes a balance between a *perfect description of the training data* and *the ability to generalize to unseen data*
- The authors propose to enforce the structural simplicity of the classification boundary by regularizing over its *topological complexity*
- The measurement of topological complexity incorporates the importance of topological features (e.g., connected components, handles, and so on), and provides a direct control over spurious topological structures
- They incorporate the new measurement as a topological penalty in training classifiers
- They also propose an algorithm to compute the *partial derivatives* (gradient) of such penalty
- They demonstrate the effectiveness of the topological regularizer on synthetic and real-world datasets

Regularizer

- A common intuition for regularization is the "Occam's razor" principle, where a regularizer enforces certain simplicity of the model in order to avoid overfitting.
- Classic regularization techniques include functional norms (L_1 or L_2 norm)

Regularizer

- A common intuition for regularization is the "Occam's razor" principle, where a regularizer enforces certain simplicity of the model in order to avoid overfitting.
- Classic regularization techniques include functional norms (L_1 or L_2 norm)
- A particularly interesting category of methods is inspired by the geometry: design new penalty terms to enforce a geometric simplicity of the classifier (e.g., enforce the classification boundary to be smooth)

Regularizer

- A common intuition for regularization is the "Occam's razor" principle, where a regularizer enforces certain simplicity of the model in order to avoid overfitting.
- Classic regularization techniques include functional norms (L_1 or L_2 norm)
- A particularly interesting category of methods is inspired by the geometry: design new penalty terms to enforce a geometric simplicity of the classifier (e.g., enforce the classification boundary to be smooth)
- But as claimed by the authors, the geometric regularizers are "structure agnostic"

Binary classification

- To demonstrate the idea, we will focus on **binary classification** and its **classification boundary**

Binary classification

- To demonstrate the idea, we will focus on **binary classification** and its **classification boundary**
- Suppose your feature is d dimensional, then a binary classifier is a function $f : \mathbb{R}^d \to \mathbb{R}$

Binary classification

- To demonstrate the idea, we will focus on **binary classification** and its **classification boundary**
- Suppose your feature is d dimensional, then a binary classifier is a function $f : \mathbb{R}^d \to \mathbb{R}$
- $f(x) \geq 0$ means x is in one class (e.g., "yes") and $f(x) < 0$ means x is in another class (e.g., "no")

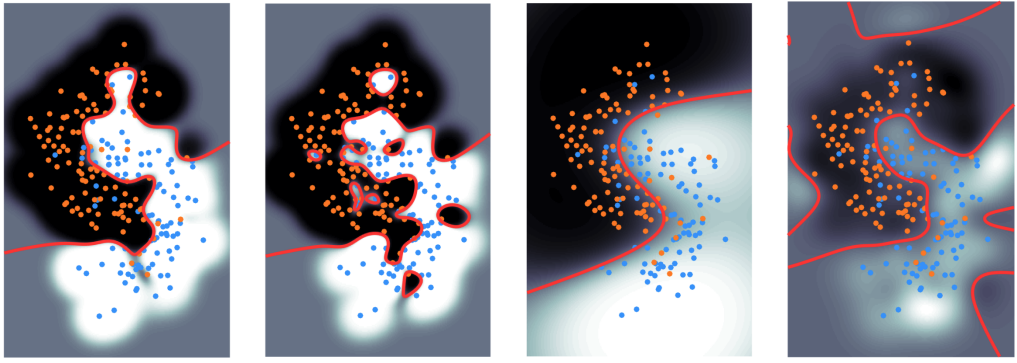
Binary classification

- To demonstrate the idea, we will focus on **binary classification** and its **classification boundary**
- Suppose your feature is d dimensional, then a binary classifier is a function $f : \mathbb{R}^d \to \mathbb{R}$
- $f(x) \geq 0$ means x is in one class (e.g., "yes") and $f(x) < 0$ means x is in another class (e.g., "no")
- $f(x) = 0$ gives the **classification boundary** (the boundary between the "yes" and "no" regions)

Binary classification

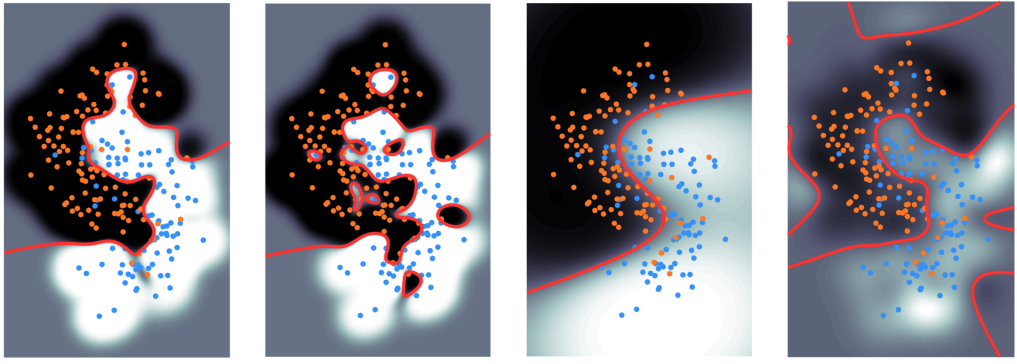
- To demonstrate the idea, we will focus on **binary classification** and its **classification boundary**
- Suppose your feature is d dimensional, then a binary classifier is a function $f : \mathbb{R}^d \to \mathbb{R}$
- $f(x) \geq 0$ means x is in one class (e.g., "yes") and $f(x) < 0$ means x is in another class (e.g., "no")
- $f(x) = 0$ gives the **classification boundary** (the boundary between the "yes" and "no" regions)
- Here, a regularizer aims at reducing the complexity of the classification boundary

Problem with geometric regularizer



(a) (b) (c) (d)

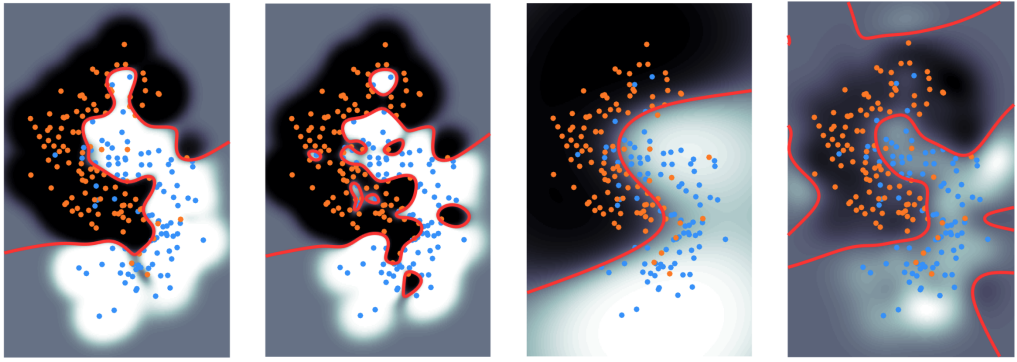
Problem with geometric regularizer



(a) (b) (c) (d)

- (b) Overfitting to data without regularizer (aka. only minimize the empirical risk)

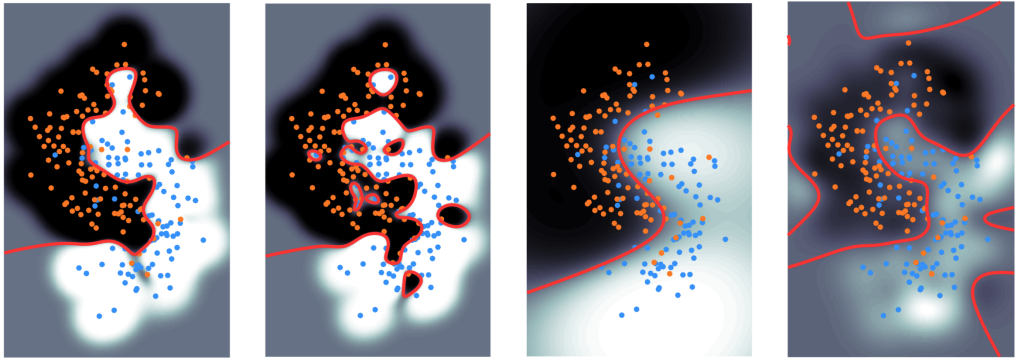
Problem with geometric regularizer



(a) (b) (c) (d)

- (b) Overfitting to data without regularizer (aka. only minimize the empirical risk)
- (c) Geometry of model (smoothness of the classification boundary) is minimized

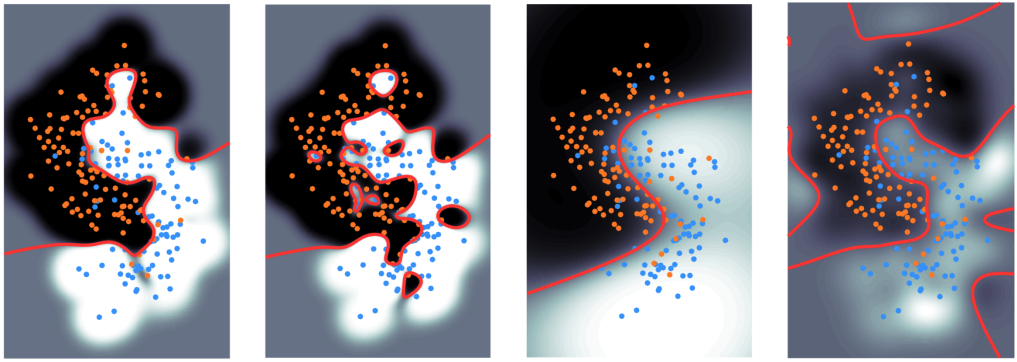
Problem with geometric regularizer



(a) (b) (c) (d)

- (b) Overfitting to data without regularizer (aka. only minimize the empirical risk)
- (c) Geometry of model (smoothness of the classification boundary) is minimized
- (d) An actual classification boundary achieved with a geometric regularizer by [Bai, Q., Rosenberg, S., Wu, Z., and Sclaroff, S. (2016). Differential geometric regularization for supervised learning of classifiers.]

Advantage of topological regularizer



(a) (b) (c) (d)

- (a) By reducing the topological complexity of the classification boundary (in this case, the no. of connected components), we achieve structural simplicity without over-smoothing the classifier boundary

Idea

- Reduce the no. of connected components (0-dimensional topological feature) in the classification boundary
- We can also do everything for higher dimensional topological features
- But everything is demonstrated for 0-dimensional feature

Idea

- Reduce the no. of connected components (0-dimensional topological feature) in the classification boundary
- We can also do everything for higher dimensional topological features
- But everything is demonstrated for 0-dimensional feature
- Formally, the classification boundary is called the **0-level set** of f:

$$f^{-1}(0) = \{x \in \mathbb{R}^d \mid f(x) = 0\}$$

Idea

- Reduce the no. of connected components (0-dimensional topological feature) in the classification boundary
- We can also do everything for higher dimensional topological features
- But everything is demonstrated for 0-dimensional feature
- Formally, the classification boundary is called the **0-level set** of f:

$$f^{-1}(0) = \{x \in \mathbb{R}^d \mid f(x) = 0\}$$

- Observe: persistent homology provides a way of counting the number of connected components of $f^{-1}(0)$ and also measure the "robustness" of each connected component

Idea

- Reduce the no. of connected components (0-dimensional topological feature) in the classification boundary
- We can also do everything for higher dimensional topological features
- But everything is demonstrated for 0-dimensional feature
- Formally, the classification boundary is called the **0-level set** of f:

$$f^{-1}(0) = \{x \in \mathbb{R}^d \mid f(x) = 0\}$$

- Observe: persistent homology provides a way of counting the number of connected components of $f^{-1}(0)$ and also measure the "robustness" of each connected component
- The "robustness" is roughly how easy it is to change the function value of f to make a component disappear

Tracking the connected components of $f^{-1}(0)$ using persistent homology

Theorem

- Let Π_f be the intervals in the 0-th persistence barcode of (sublevelset filtration of) f **containing value 0**

Tracking the connected components of $f^{-1}(0)$ using persistent homology

Theorem

- Let Π_f be the intervals in the 0-th persistence barcode of (sublevelset filtration of) f **containing value 0**
- Let Π_{-f} be similarly defined

Tracking the connected components of $f^{-1}(0)$ using persistent homology

Theorem

- Let Π_f be the intervals in the 0-th persistence barcode of (sublevelset filtration of) f **containing value 0**
- Let Π_{-f} be similarly defined
- There is a one-to-one correspondence between:
  - all but one connected components in $f^{-1}(0)$
  - and $\Pi_f \cup \Pi_{-f}$

Tracking the connected components of $f^{-1}(0)$ using persistent homology

Theorem

- Let Π_f be the intervals in the 0-th persistence barcode of (sublevelset filtration of) f **containing value 0**
- Let Π_{-f} be similarly defined
- There is a one-to-one correspondence between:
  - all but one connected components in $f^{-1}(0)$
  - and $\Pi_f \cup \Pi_{-f}$

Above theorem implies:

- Each interval $\Pi_f \cup \Pi_{-f}$ tracks a connected component in $f^{-1}(0)$
- We could thus track almost all connected components in $f^{-1}(0)$ except one

Tracking the connected components of $f^{-1}(0)$ using persistent homology

Theorem

- Let Π_f be the intervals in the 0-th persistence barcode of (sublevelset filtration of) f **containing value 0**
- Let Π_{-f} be similarly defined
- There is a one-to-one correspondence between:
  - all but one connected components in $f^{-1}(0)$
  - and $\Pi_f \cup \Pi_{-f}$

Above theorem implies:

- Each interval $\Pi_f \cup \Pi_{-f}$ tracks a connected component in $f^{-1}(0)$
- We could thus track almost all connected components in $f^{-1}(0)$ except one
- But this doesn't matter: we should always allow at least one connected component in $f^{-1}(0)$

Tracking the connected components of $f^{-1}(0)$ using persistent homology

Theorem

- Let Π_f be the intervals in the 0-th persistence barcode of (sublevelset filtration of) f **containing value 0**
- Let Π_{-f} be similarly defined
- There is a one-to-one correspondence between:
  - all but one connected components in $f^{-1}(0)$
  - and $\Pi_f \cup \Pi_{-f}$

Above theorem implies:

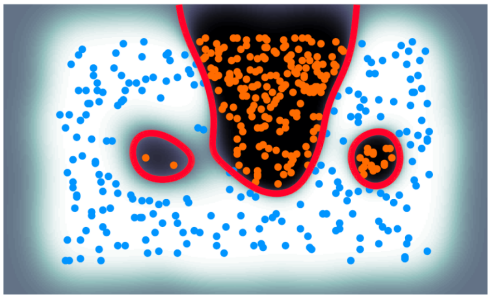
- Each interval $\Pi_f \cup \Pi_{-f}$ tracks a connected component in $f^{-1}(0)$
- We could thus track almost all connected components in $f^{-1}(0)$ except one
- But this doesn't matter: we should always allow at least one connected component in $f^{-1}(0)$
- So as long as we are able to track all other connected components in $f^{-1}(0)$ and have a way of measuring their "robustness" and make those less robust ones disappear, we are good

Measuring the robustness of a connected component in $f^{-1}(0)$

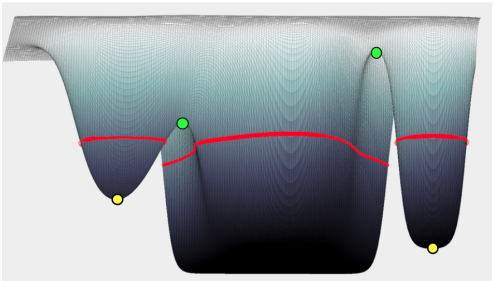
- For an interval $[b, d) \in \Pi_f$: b is a local minimum of f below 0 creating a component, and d is a saddle above 0 merging two components
- We have similar facts for intervals in Π_{-f}

Measuring the robustness of a connected component in $f^{-1}(0)$

- For an interval $[b, d) \in \Pi_f$: b is a local minimum of f below 0 creating a component, and d is a saddle above 0 merging two components
- We have similar facts for intervals in Π_{-f}



(a)



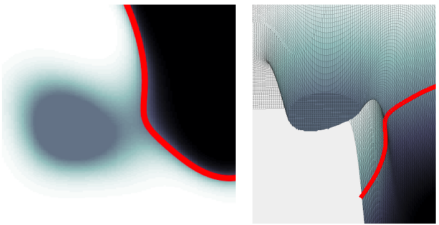
(b)

Measuring the robustness of a connected component in $f^{-1}(0)$

- We have that for the interval $[b, d) \in \Pi_f$, b and d measures how hard it is to remove the connected component in $f^{-1}(0)$

Measuring the robustness of a connected component in $f^{-1}(0)$

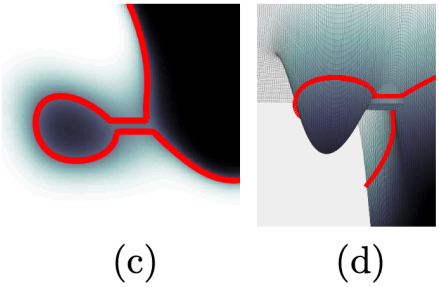
- We have that for the interval $[b, d) \in \Pi_f$, b and d measures how hard it is to remove the connected component in $f^{-1}(0)$
- If we "push" the birth (local minimum) up, the left component disappears:



(a) (b)

Measuring the robustness of a connected component in $f^{-1}(0)$

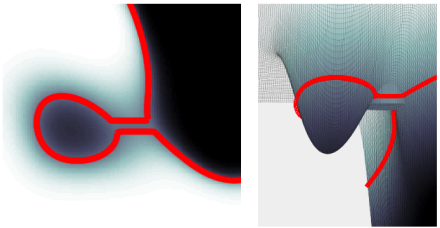
- We have that for the interval $[b, d) \in \Pi_f$, b and d measures how hard it is to remove the connected component in $f^{-1}(0)$
- If we "pull" the death (saddle) down, the left component merges with the right one:



(c) (d)

Measuring the robustness of a connected component in $f^{-1}(0)$

- We have that for the interval $[b, d) \in \Pi_f$, b and d measures how hard it is to remove the connected component in $f^{-1}(0)$
- If we "pull" the death (saddle) down, the left component merges with the right one:



(c) (d)

- So the robustness of the connected component (min. effort to remove it) is:

$$\min\{|b|, |d|\}$$

Topological penalty of the classifier

- The topological penalty of the classifier is then sum of the squared robustness of each interval (connected component) in $\Pi_f \cup \Pi_{-f}$, excluding the one with the maximum robustness (we should always ensure there is at least one major component in the classification boundary)

$$\mathcal{R}_T(f) = \sum_{[b,d) \in \Pi_f \cup \Pi_{-f} \setminus \max(\Pi_f \cup \Pi_{-f})} (\min\{|b|, |d|\})^2$$

Computing the partial derivatives for $\mathcal{R}_T(f)$

- Computing partial derivatives exactly for $\mathcal{R}_T(f)$ can be hard as the function f itself can contain a lot of local min., local max., or saddles

Computing the partial derivatives for $\mathcal{R}_T(f)$

- Computing partial derivatives exactly for $\mathcal{R}_T(f)$ can be hard as the function f itself can contain a lot of local min., local max., or saddles
- They try to discretize the domain $\mathbb{R}^d$ (essentially a cube by making it bounded) into graphs (e.g., regular grids), and them compute the PD on the discretized domain

Computing the partial derivatives for $\mathcal{R}_T(f)$

- Computing partial derivatives exactly for $\mathcal{R}_T(f)$ can be hard as the function f itself can contain a lot of local min., local max., or saddles
- They try to discretize the domain $\mathbb{R}^d$ (essentially a cube by making it bounded) into graphs (e.g., regular grids), and them compute the PD on the discretized domain
- They then show that $\mathcal{R}_T(f)$ is *differentiable almost everywhere* (except some isolated points), which should be good enough for gradient descent

Experiments

- combine it with a kernel logistic regression classifier to demonstrate its advantage

Experiments

- combine it with a kernel logistic regression classifier to demonstrate its advantage
- compare their method with several baselines: k-nearest-neighbor classifier (KNN), logistic regression (LG), Support Vector Machine (SVM), and Kernel Logistic Regression (KLR) with functional norms (L_1 and L_2) as regularizers

Experiments

- combine it with a kernel logistic regression classifier to demonstrate its advantage
- compare their method with several baselines: k-nearest-neighbor classifier (KNN), logistic regression (LG), Support Vector Machine (SVM), and Kernel Logistic Regression (KLR) with functional norms (L_1 and L_2) as regularizers
- also compare with two state-of-the-art methods based on geometric regularizers: the Eulers Elastica classifier (EE) and the Classifier with Differential Geometric Regularization (DGR)

Experiments

- combine it with a kernel logistic regression classifier to demonstrate its advantage
- compare their method with several baselines: k-nearest-neighbor classifier (KNN), logistic regression (LG), Support Vector Machine (SVM), and Kernel Logistic Regression (KLR) with functional norms (L_1 and L_2) as regularizers
- also compare with two state-of-the-art methods based on geometric regularizers: the Eulers Elastica classifier (EE) and the Classifier with Differential Geometric Regularization (DGR)
- In order to thoroughly evaluate the behavior of the model, especially in large noise regime, created synthetic data with various noise levels: beside feature space noise, also inject different levels of label noise, e.g., randomly perturb labels of 0%, 5%, 10% and 20% of the training data
- also evaluate their method on real world data

Synthetic							
	KNN	LG	SVM	EE	DGR	KLR	TopoReg
Blob-2 (500,5)	7.61	8.20	7.61	8.41	7.41	7.80	**7.20**
Moons (500,2)	20.62	20.00	19.80	19.00	19.01	18.83	**18.63**
Moons (1000,2,Noise 0%)	19.30	19.59	19.89	17.90	19.20	17.80	**17.60**
Moons (1000,2,Noise 5%)	21.60	19.29	19.59	22.00	22.30	**19.00**	**19.00**
Moons (1000,2,Noise 10%)	21.10	**19.19**	19.89	24.40	26.30	20.00	19.70
Moons (1000,2,Noise 20%)	23.00	19.79	**19.40**	30.60	30.20	19.50	**19.40**
AVERAGE	18.87	17.68	17.70	20.39	20.74	21.63	**16.92**
UCI							
	KNN	LG	SVM	EE	DGR	KLR	TopoReg
SPECT (267,22)	17.57	17.20	18.68	**16.38**	23.92	18.31	17.54
Congress (435,16)	5.04	4.13	4.59	4.59	4.80	**4.12**	4.58
Molec. (106,57)	24.54	19.10	19.79	17.25	16.32	19.10	**12.62**
Cancer (286,9)	29.36	28.65	28.64	28.68	31.42	29.00	**28.31**
Vertebral (310,6)	15.47	15.46	23.23	17.15	13.56	12.56	**12.24**
Energy (768,8)	0.78	0.65	0.65	0.91	0.78	**0.52**	**0.52**
AVERAGE	15.46	14.20	15.93	14.16	15.13	13.94	**11.80**
Biomedicine							
	KNN	LG	SVM	EE	DGR	KLR	TopoReg
KIRC (243,166)	30.12	28.87	32.56	31.38	35.50	31.38	**26.81**
fMRI (1092,19)	46.70	74.91	74.08	82.51	**31.32**	34.07	33.24