Classification Based On Topological Data Analysis

Aina Ferrà Marcús

TML@UB

January 14, 2022

The classification problem

Classification is the problem of identifying to which subcategory an observation belongs to.

It is one of the most common and treated problems in Machine Learning. Some examples of classifiers are the simplest NN (**N**earest **N**eighbour) to the more complex NN (**N**eural **N**etworks).

Usually one has a set of the data with known labels (**training set**) that can be used to fine tune a classifier. The algorithm is used then with another set of data to which the labels are supposed unkown (**test set**) to assess the performance.

Classification in Topological Data Analysis

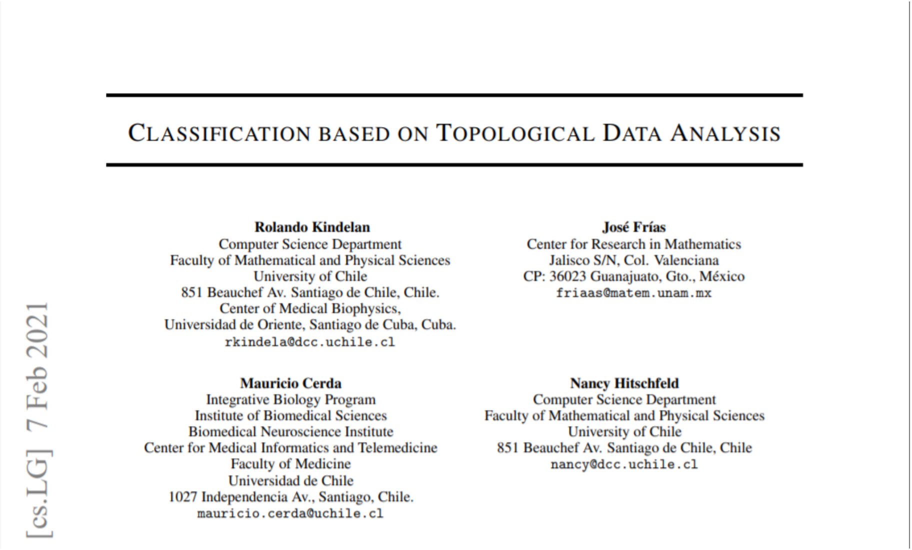
Topological Data Analysis has been used mainly to obtain descriptors of data but not to intrinsically classify.

TDA techniques, including:

- Mapper (qualitative information)
- Persistence diagrams and persistence images (2D representations)
- Betti curves and landscapes (1D vector-like representations)
- Other persistence summaries (statistical information)

often **offer support to Machine Learning algorithms** in order to classify, but there is still room to improve classification solely based on topological processes.

A first topological based classifier



Classification based on Topological Data Analysis

Rolando Kindelan
Computer Science Department
Faculty of Mathematical and Physical Sciences
University of Chile
851 Beauchef Av. Santiago de Chile, Chile.
Center of Medical Biophysics,
Universidad de Oriente, Santiago de Cuba, Cuba.
rkindela@dcc.uchile.cl

José Frías
Center for Research in Mathematics
Jalisco S/N, Col. Valenciana
CP: 36023 Guanajuato, Gto., México
friaas@matem.unam.mx

Mauricio Cerda
Integrative Biology Program
Institute of Biomedical Sciences
Biomedical Neuroscience Institute
Center for Medical Informatics and Telemedicine
Faculty of Medicine
Universidad de Chile
1027 Independencia Av., Santiago, Chile.
mauricio.cerda@uchile.cl

Nancy Hitschfeld
Computer Science Department
Faculty of Mathematical and Physical Sciences
University of Chile
851 Beauchef Av. Santiago de Chile, Chile
nancy@dcc.uchile.cl

Outline of the presentation

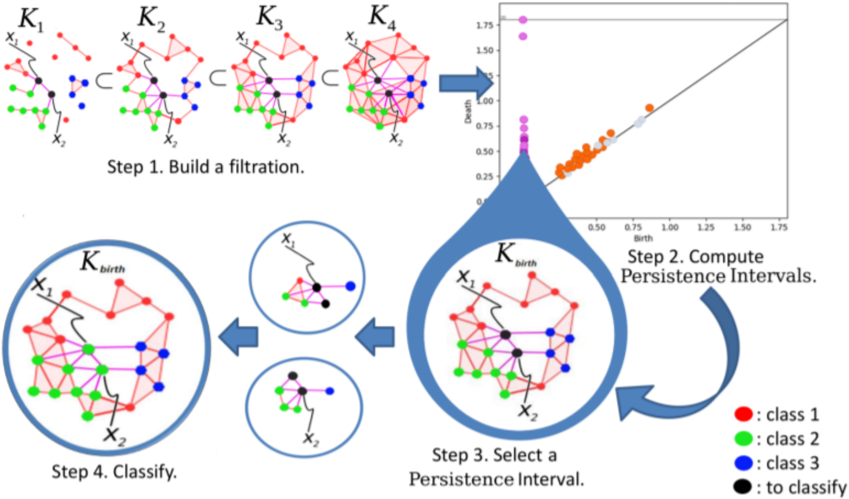
1. **Overview of the algorithm**. Focusing on the three main steps, as described on the paper.
2. **Brief review of the results**.
3. **Discussion, free comments, brainstorming**. Anything that can be useful for this seminar.

Overview of the algorithm

The algorithm aims to apply TDA to a multi-class dataset without **any further use of Machine Learning**. The steps are as follows:

1. Build a filtration on the **whole dataset** of training and test data and apply persistence homology.
2. Analyze results to choose **the best possible sub-complex** to apply classification.
3. For each test sample, assign a label based on their **link neighbourhood** labels.

Overview of the algorithm



Step 1. Build a filtration.

Step 2. Compute Persistence Intervals.

Step 3. Select a Persistence Interval.

Step 4. Classify.

● : class 1

● : class 2

● : class 3

● : to classify

1. Building a simplicial complex

The first step of the algorithm consists on building and analyzing a filtration on the **whole dataset of training and test data**.

The idea is to find the **underlying shape of the whole dataset**, so test samples must contribute to that.

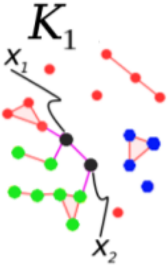
Technical comment

In the original paper all computations are done in `GUDHI`, but since this step needs only the results of the persistence homology, could be done with other libraries.

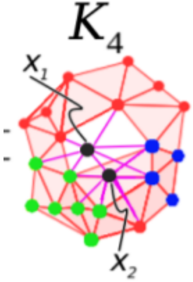
2. Choosing a sub-complex

Idea

We want to look at the neighbourhood of each test sample to classify. **The neighbourhood will be defined by a sub-complex in the filtration**. Which sub-complex?



(a) Too many isolated points.



(b) Too much connectivity.

2. Choosing a sub-complex

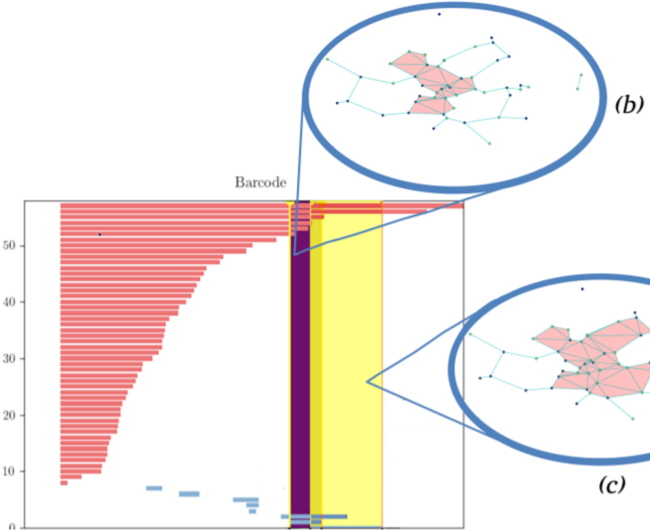
1. The paper proposes three strategies to choose a **persistence interval**.

   - The **maximum** persistence interval except the infinite ray (TDABC-M).
   - A persistence interval selected in a **random** way (TDABC-R).
   - The persistence interval closest to the **average** persistence interval (TDABC-A).

2. A persistence interval is an interval defined by parameters (birth, death). The **chosen sub-complex** will be the complex in the filtration **corresponding to the birth** parameter.

   The authors mention that the sub-complexes corresponding to the *death parameter* and *(birth + death)/2* parameter could be selected, but do not elaborate.

2. Choosing a sub-complex



2. Choosing a sub-complex

Once one parameter of the filtration has been selected, the corresponding sub-complex is built and will be used to classify.

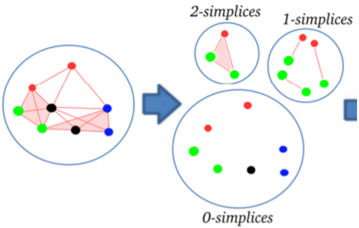
Technical comment

In this step of the algorithm we need to actually **compute and maintain in memory the chosen sub-complex**. Computations in the original paper are done using `GUDHI`, and although this library can be changed, we need to use a library that allows us to access to the actual simplicial complex.

3. Classification

Classification for a test sample is based on a neighbourhood so we need to define what does **neighbourhood** mean here. A couple of definitions:

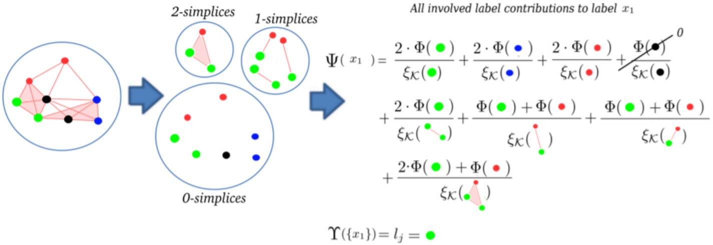
- The **star of a vertex** in a simplicial complex is the set of all simplices that contain the given vertex.
- The **closed star** of a vertex is the smallest simplicial complex that contains the star of the vertex.
- The **link of a vertex** is the set of simplices in its closed star that do not share any face with the vertex.



3. Classification

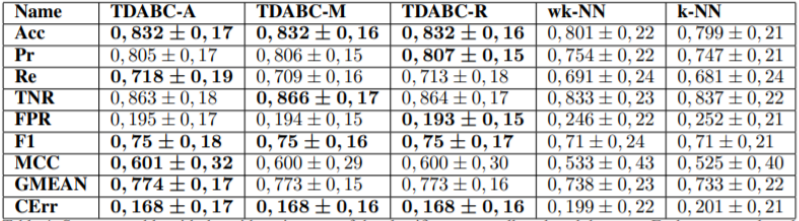
Two important ideas behind weighting the labels in the neighbourhood:

1. **Elements that cluster around the test sample earlier should contribute more to the classification** (proximity, similarity).
2. **Elements that appear to several dimensional simplices should contribute more to the classification** (importance by topological features).



Results

The algorithm is tested for 5 synthetic datasets (Circles, Moon, Swissroll, Normdist, Sphere) and 3 real world datasets (Breast cancer, Wine, Iris).



Name	**TDABC-A**	**TDABC-M**	**TDABC-R**	**wk-NN**	**k-NN**
Acc	**0, 832 ± 0, 17**	**0, 832 ± 0, 16**	**0, 832 ± 0, 16**	0, 801 ± 0, 22	0, 799 ± 0, 21
Pr	0, 805 ± 0, 17	0, 806 ± 0, 15	**0, 807 ± 0, 15**	0, 754 ± 0, 22	0, 747 ± 0, 21
Re	**0, 718 ± 0, 19**	0, 709 ± 0, 16	0, 713 ± 0, 18	0, 691 ± 0, 24	0, 681 ± 0, 24
TNR	0, 863 ± 0, 18	**0, 866 ± 0, 17**	0, 864 ± 0, 17	0, 833 ± 0, 23	0, 837 ± 0, 22
FPR	0, 195 ± 0, 17	0, 194 ± 0, 15	**0, 193 ± 0, 15**	0, 246 ± 0, 22	0, 252 ± 0, 21
F1	**0, 75 ± 0, 18**	**0, 75 ± 0, 16**	**0, 75 ± 0, 17**	0, 71 ± 0, 24	0, 71 ± 0, 21
MCC	**0, 601 ± 0, 32**	0, 600 ± 0, 29	0, 600 ± 0, 30	0, 533 ± 0, 43	0, 525 ± 0, 40
GMEAN	**0, 774 ± 0, 17**	0, 773 ± 0, 15	0, 773 ± 0, 16	0, 738 ± 0, 23	0, 733 ± 0, 22
CErr	**0, 168 ± 0, 17**	**0, 168 ± 0, 16**	**0, 168 ± 0, 16**	0, 199 ± 0, 22	0, 201 ± 0, 21

Table 4: Summary table with the arithmetic mean of the classifiers across all analyzed data sets. Each mean result and standard deviation is shown for the performance in this metric across all datasets.

Results are consistently similar to the weighted k-NN and the k-NN algorithms, only slightly better in some datasets.

Discussion

Some topics are not properly elaborated on the paper and there is still room for improvement.

1. When analyzing the whole dataset, **how many dimensions of persistence** should we look at? Is performance affected by it?
2. Why are the heuristics proposed in this paper to choose a sub-complex good? Can we **try all possible sub-complexes** and perform optimization?
3. **How many dimensions** should we consider in a sub-complex to compute the link of a vertex?
4. What happens with **non-useful simplices**? What happens with **isolated points**?

Thank you for your attention. Let's discuss!