Cosine Similarity based TopoMap

Devarsh Patel
dp3324@nyu.edu

Master of Science in Computer Science,
NYU Tandon School of Engineering

May 7, 2022

Track: Track 1

Github: https://github.com/dp3324/advance-topomap

Abstract

The main goal of this project is to take TopoMap[2] one step further in terms of information computation complexity and space efficiency. It will answer many unresolved bottlenecks emerged during the initial iteration in the previous work. The point of interest for designing and developing this advanced TopoMap[2] solution is to answer the training problem that arise in Machine Learning models because of High-Dimensional data. While using polynomial data with multiple regression model, It requires more computation time along with intensive space utilization. This project aims to minimize the dimension of the data while preserving the model accuracy and precision. The advancement of this technology can help data engineers to identify and modify the data cloud as per the requirement so that they can focus on whats more relevant that something that is unimportant.

1 Objective

The objective of this research paper is to optimize the conventional TopoMap[2] algorithms so that it can efficiently define the outliners, provide better point deviation in the final representation, improve the relationship between points and their clusters.

2 Introduction

TopoMap[2] is an advanced Multidimensional Projection (MDP) technique used for high-dimensional data, that ensures to maintain the topographical definition of data. Dealing with high-dimensional data create complex computation intensity with the increase size in dataset. To over-come this bottleneck, it is always preferred to only consider the dimension, i.e. parameters, of the dataset which has the most impact on the predicted output. This means it is more suitable to remove inert parameters or dimension of the dataset to minimize the computation time. Here where TopoMap[2] comes, It provides better visualizations of introduced data by decreasing d-dimensional data to 2D Space vector. TopoMap[2] is an advanced implementation of sophisticated projection techniques like Classical MDS, IsoMap, tSNE and UMAP. Most of the Multidimensional projection technique uses geographical distance as a base to define the relationship between the data points, which often results into false clustering as many times the data points in same coordinate region have significant topographical variance. Multidimensional scaling(MDS) is term coined for techniques which reduces data dimension by significant proportion. In context of projection, if a technique reduces data dimension to a point where it can be represented in 2D or 3D cartesian space, then that is called as Multidimensional Projection (MDP).

This research paper focuses on two main aspects of original TopoMap algorithm: Euclidian Minimum Spanning Tree, and Convex Hull Alignment Algorithm. The proposed technique here uses Cosine Similarity based Minimum Spanning Tree instead of Euclidian distance. To accommodate the changes and the data type, Tree based hull alignment algorithm is proposed here. This algorithm varies from the convex hull alignment algorithm in terms of point placement in the component.

3 Literature Survey

There are many data dimensionality reduction algorithm designed to preserve the relationship between the points but most to them are based on the parameters dependents on the geographical terms. They utilizes distance based matrices like Euclidian distance to define the co-relation. They all are efficient in their own terms but comes with many limitations like leaving n-level simplexes. Many of this algorithms certainty decreases as the size of the data increases.

3.1 Multidimensional Scaling

Multidimensional Scaling or MDS is the simplest means of visualizing the similarities between the cases in the dataset. MDS uses the distance between n objects and define the cluster based on them. MDS places the n-dimensional data in 2D cartesian space such that it maintains the distance between the datapoints. There are mainly 4 type of Multidimensional Scaling algorithms.

3.1.1 Classical Multidimensional scaling

Classical multidimensional scaling also known as Principle Component Analysis (PCA), Torgerson Scaling or Torgerson–Gower scaling, is an algorithm which takes the input matrix giving the dissimilarities between the pairs of items and place them such that the loss function or strain is minimized. The strain can be defines as follows:

$$Strain_D(x_1, x_2....x_n) =$$

$$\left(\frac{\sum_{i,j}(b_{ij} - x_i^T x_j)^2}{\sum_{i,j} b_{ij}^2}\right)^{1/2}$$

where x_i are the N-dimensional space vectors, $x_i^T x_j$ defines the scalar product between x_i and x_j and b_{ij} are the element of matrix B.

3.1.2 Metric Multidimensional scaling

Metric Multidimensional scaling is similar to the Classical Multidimensional scaling but it tries to minimize the Strain in the space instead of the stress.

$$Stress_D(x_1, x_2....x_n) =$$

$$\sqrt{\sum_{i \neq j=1,..,N} (d_{ij} - \|x_i - x_j\|)^2}$$

3.1.3 Non-metric multidimensional scaling

In contrast to non classical and metric multidimensional scaling, Non-metric MDS finds both the non-parametric monotonic between the data points and the Euclidian distance between them. This data dimensionality reduction algorithm is based on isotonic regression.

For x denote the vector of proximities, $f(x)$ a monotonic transformation of x, and d as the point distance, then the coordinate of the points in 2D cartesian are found by minimizing the stress as follows:

$$Stress = \sqrt{\frac{\sum (f(x) - d)^2}{\sum d^2}}$$

3.1.4 Generalized multidimensional matrix

This method is extension of metric multidimensional scaling which utilizes the non-Euclidian space. It allows finding the minimum-distortion embedding of one surface into another.

3.2 Topology-based Multidimensional Projection

Topology-based methods exist from a very long time when it comes to visualizing data. It has played an important role in many visualization prone domains like astrophysics, medical imaging, robotics and material science. The described TopoMap is based on the Rips[1] filtration technique. Rips filtration technique is used to topographically analysis the high-dimensional data point which also efficiently capture the homology of the manifold sampled by this data points. It also correlates to the Reeb Graph[4], which contract the cluster of similar topological definition to a single point, which result into skeleton like visualization. When it comes to discrete point cloud visualization, the Mapper[5] is an better approach which is based on Reed[4] approximation for the nearest neighbor graph.

3.3 Topology-based Multidimensional Projection

TopoMap[2] provides an in-depth comparison to multiple Multidimensional Scaling(MDP)/Multidimensional Projection(MDP) techniques. While most of them works on geodesic distance between the data points, especially conventional Isomap, TopoMap[2] works on homological difference between the points. IsoMap, tSNE and UMAP have many improved variation but they all does not preserve 0-cyclic groups. An Isomap variant proposed by Lee and Verleysen[3], tears the high-dimension data into non-contractable loops which preserves manifold unfolding.

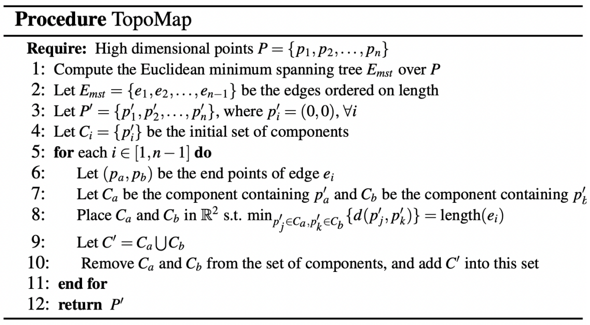
The work represented here, by considering Rips Filtration[1], is focused on analyzing topographical distance. The TopoMap is compared to the terrain metaphors by the separation point of preserving the topological angle. It is guarantied that TopoMap preserves the connected component during filtration and result into identical 0-cyclic homology persistence diagram from the produced result and original data.

4 Methodology

The methodology proposed here is an extension to the original implementation of TopoMap[2] to improve outliner detection, clear point deviation in the cluster, and relations between points and clusters. This focused areas in the original TopoMap[2] are: Minimum Spanning Tree and Hull Alignment algorithm.

4.1 TopoMap: Original Algorithm

This original TopMap[2] was implemented using Euclidian Minimum Spanning Tree and Convex Hull Alignment. It was designed and developed in C++. The overall time-complexity of this algorithm is $\mathrm{O}N \log N \alpha N$. This algorithm was divided in 2 phases: Calculate Euclidian Minimum Spanning Tree and Align components based on Convex Hull Algorithm.



```
Procedure TopoMap
Require: High dimensional points $P = \{p_1, p_2, \ldots, p_n\}$
 1: Compute the Euclidean minimum spanning tree $E_{mst}$ over $P$
 2: Let $E_{mst} = \{e_1, e_2, \ldots, e_{n-1}\}$ be the edges ordered on length
 3: Let $P' = \{p'_1, p'_2, \ldots, p'_n\}$, where $p'_i = (0,0), \forall i$
 4: Let $C_i = \{p'_i\}$ be the initial set of components
 5: for each $i \in [1, n-1]$ do
 6:     Let $(p_a, p_b)$ be the end points of edge $e_i$
 7:     Let $C_a$ be the component containing $p'_a$ and $C_b$ be the component containing $p'_b$
 8:     Place $C_a$ and $C_b$ in $\mathbb{R}^2$ s.t. $\min_{p'_j \in C_a, p'_k \in C_b} \{d(p'_j, p'_k)\} = \text{length}(e_i)$
 9:     Let $C' = C_a \bigcup C_b$
10:     Remove $C_a$ and $C_b$ from the set of components, and add $C'$ into this set
11: end for
12: return $P'$
```

Figure 1: Original TopoMap Implementation

4.2 Minimum Spanning Tree: Cosine Similarity algorithm

As the original TopoMap[2] uses Euclidian distance as the base of creating Minimum Spanning Tree, It only defines the geometrical, specifically distance, relations between the data. While this method is better, it does not take point inclination into the account. The Cosine Similarity is the well know method used in Natural Language Processing, to find the similarities between two or more words. Taking the inference from this, This paper proposed changes to the TopoMap[2] implementation by designing it using the radian angle value instead of euclidian distance. The modified algorithm is stated in Figure 3.

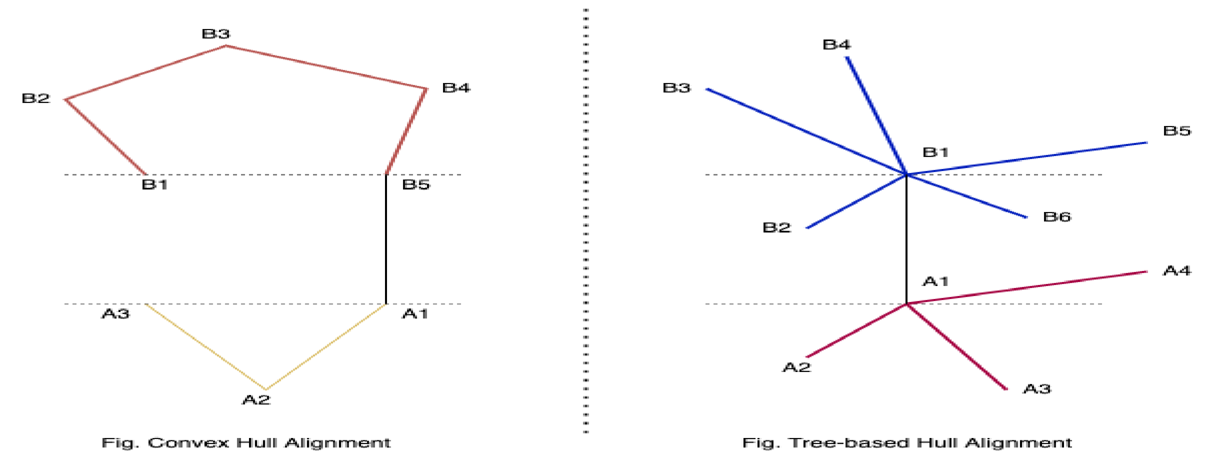


Figure 2: Comparison between Convex Hull Alignment and Tree-based Hull Alignment algorithm.

```
Vnew[] = {x}
Enew[] = {}
while Vnew is not equal to V
    u -> a node from Vnew
    v -> a node that is not in Vnew such that
         edge u—v has the angle between them
    add v to Vnew
    add edge (u, v) to Enew
end while
Return Vnew and Enew
```

Figure 3: Cosine Similarity based Minimum Spanning Tree

4.3 Tree-based hull alignment algorithm

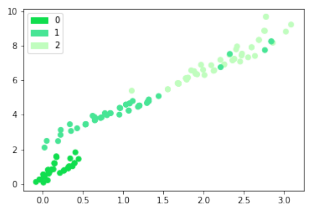
The most important part of the original TopoMap[2] implementation is to plot the minimum spanning tree obtained using hull alignment algorithm. These algorithm takes two points from different components containing the edge vertices and place them such that one component is shaped convex and the other one as concave as showed on left side of Figure 2.Even though this provided better visualization it does not provide proper point deviation. This method can run into serious alignment issues if the size of the components increases. It can run out of joining sides. Also it should be taken into consideration that this algorithm only uses the points on the outer edge of the component to align the hull. That means that the components might not joined on the edge used. Which in turns remove the original relationship between the points and also between their clusters. While this method is useful, It can be improved if the algorithm take all the points in the component into consideration to align the hull. The proposed tree-based hull alignment algorithm uses the sum of y-values of the points in the component to decide the weight. to alight the component in the convex fashion, the algorithm minimize the weight and vice-versa for the concave fashion. The result of this improved algorithm can be seen on right side of Figure 4. The pseudo-code of this algorithm is shown in Figure 2.

```
Cset = [{P1}, {P2},...{Pi}]
Pa = point of interest in Cset
weight = null
for Pi in Cset:
  Pi = Pi − Pa \\ subtract point of interest
                  from each point
make weight as max(concave) or min(convex):
  updated_component, weight = rotate(component) \\ rotate by 1 deg
                                                    in each iteration
for Pi in Cset:
  Pi = Pi + Pa \\ get the original coordinates back
return updated_component
```

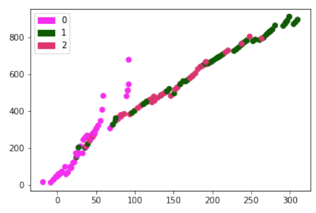
Figure 4: Tree-based Hull Alignment Algorithm

This improved algorithm was not only able to provide better relations between the point but was also able to create improved point deviation among the points and clusters. The propose tree based hull alignment was clearly and efficiently was able to provide the relation between the points and their cluster.

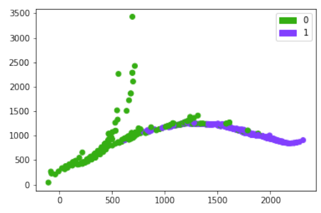
The time complexity was decreased by approximately 25% as the space-complexity of the algorithm decreased by significant amount of approximately 10% by increasing the complexity of calculations due to Cosine Similarity based Euclidian Minimum Spanning Tree and Tree-based hull alignment algorithm.



(a) Iris Dataset

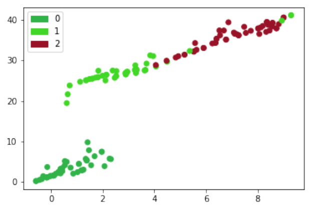


(b) Wine Dataset

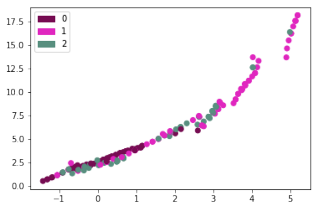


(c) Breast Cancer Dataset

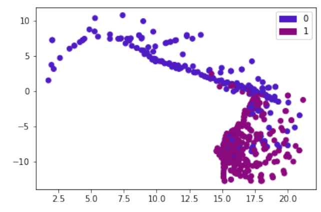
Figure 5: Euclidian Distance based TopoMap



(a) Iris Dataset



(b) Wine Dataset



(c) Breast Cancer Dataset

Figure 6: Cosine Similarity based TopoMap

5 Results & Discussion

Topological representation of the high dimensional data is a better option compared to the convention multidimensional scaling algorithms like Multidimensional Scaling, T-SNE, and UMAP. The algorithm proposed here was efficient optimized, scalable, and robust. This improvised algorithm of the original TopoMap[2] implementation provided solid proofs of data relationship preservation of both 0-cyclic and correlation based on inclination. It was tested using open-source datasets available on Scikit learn library like Iris, Wine, and Breast Cancer.

Data set	# Inst.	# Attr.	# Cls.
Iris	150	5	3
Wine	178	13	3
Breast Cancer	569	30	2

As we can see in Fig. 5 and Fig. 6, that the angle based implementation is providing better point separation and deviation compared to Euclidian distance based minimum spanning tree on improved hull alignment algorithm. We can clearly notice that even when the data is clouded and intensive the algorithm was able to provide sufficient differentiation compared to original TopoMap[2] implementation.

The overall processing and running time for both of these algorithms was 10% less than the original algorithm of TopoMap[2].

6 Future Scope

There are many improvement that can be done in this improved TopoMap[2] as follows:
3-Dimensional Implementation: It will be exciting to see the plotted data in 3D, as it will provide details on covered datapoints.
Optimized MST: It will be more efficient if there is a way to optimize the current minimum spanning tree by decreasing time complexity to $O(N \log N)$.

7 Conclusion

The interpretation of N-dimensional dataset was efficiently reduced to 2D plotting and was able to increase model training speed by almost 75% which is more efficient when it comes to large datasets. Extensions of this algorithms will help in decreasing it deep learning models hyper-parameters and layer size.

References

[1] U. Bauer. “Ripser: efficient computation of vietoris-rips persistence bar- codes”. In: (Aug. 2019).

[2] Harish Doraiswamy et al. “TopoMap: A 0-dimensional Homology Preserving Projection of High-Dimensional Data”. In: *IEEE Transactions on Visualization and Computer Graphics* 27.2 (2020), pp. 561–571. DOI: `https://arxiv.org/pdf/2009.01512.pdf`.

[3] J.A.Lee and M.Verleysen. “Nonlineardimensionalityreductionofdata manifolds with essential loops”. In: *Neurocomputing* (2005), pp. 29–53.

[4] Georges Henri Reeb. “Sur les points singuliers d’une forme de Pfaff complètement intégrable ou d’une fonction numérique”. In: *C. R. Acad. Sci. Paris* (1946), pp. 847–849.

[5] G. Singh, F. Memoli, and G. Carlsson. “Topological Methods for the Analysis of High Dimensional Data Sets and 3D Object Recognition”. In: *Eurographics Symposium on Point-Based Graphicses* (2007).