Graph-Induced Geodesics Approximation for Non-Euclidian K-Means

Hervé Frezza-Buet

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

Graph-Induced Geodesics Approximation for Non-Euclidian K-Means

Hervé Frezza-Buet

Université de Lorraine, CentraleSupélec, CNRS, LORIA, F-57000 Metz, France
herve.frezza-buet@centralesupelec.fr

Abstract. In this paper, an adaptation of the k-means algorithm and related methods to non-Euclidian topology is presented. The paper introduces a rationale for approximating the geodesics of that topology, as well as a learning rule that is robust to noise. The first results on artificial but very noisy distributions presented here are promising for further experimentation on real cases.

1 Introduction

This paper is a contribution to the field of *vector quantization*. Rather than using the Euclidian distance in the sample space, our approach builds up a graph from the dataset that approximates the underlying manifold topology. A similar idea has been proposed in [1], where a k-nn graph is built, having one vertex per sample in the dataset. It is used to compute a distance matrix between all pairs of samples, based on the shortest path length in that graph from one sample to others. As the algorithm is a c-medoid rather than a k-means, it can rely only on that matrix, computed once beforehand, to find the medoids. In this paper, the graph is much smaller, since only a fraction of the samples is used as vertices. It is obtained by Competitive Hebbian Learning [2]. Moreover, we rather use k-means, which requires moving prototypes along geodesics during learning (which is the linear interpolation in the default Euclidian case). This enables to have prototypes that are not some selected samples of the dataset (they can be inbetween), but above all it opens the method to the adaptation of algorithms like Self-Organizing Maps [3] to non-Euclidian geodesics, as illustrated in previous work, since the weight updating rules of these vector quantization algorithms all consist basically in walking along geodesics.

Last, the originality of this paper is the handling of noise. Indeed, clusters may not be that formally well-defined when noisy samples can be found in the space between the components of the manifold. Despite a lack of cluster definition in that case, in practice, one would like to detect significant groupings of data samples even if the groups are not well-separated clusters of samples.

2 Graph-induced topology (GIT)

Let us denote by $S \stackrel{\text{def}}{=} \{\xi_1, \ldots, \xi_n\}$, $\xi_i \in X$ the set of samples available for vector quantization. They lie in the input space X where a distance d is available. The samples are considered as elements taken randomly from an unknown manifold $\mathcal{M} \subset X$. The manifold $\mathcal{M}$ can have a complex topology, it can contain

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

holes, parts having different dimensions, curved areas, etc. The distance d does not reflect the topology of $\mathcal{M}$, but it can be used to infer that topology from the samples in S. The trick is to compute a graph $\mathcal{G} \stackrel{\text{def}}{=} (\mathcal{V}, \mathcal{E})$ ($\mathcal{V}$ stands for the set of vertices, $\mathcal{E}$ for the edges) that covers the manifold $\mathcal{M}$, such as the topology induced by its edges reflects the structure of $\mathcal{M}$. This can be easily achieved thanks to the Competitive Hebbian Learning algorithm (CHL) introduced in [2]. This consists in the following. First, let us select randomly m samples $(\xi_1, \ldots, \xi_m)$ from S, with $m \simeq \gamma|S|$, $\gamma \in]0,\ 1[$. These selected samples enable to define each vertex v of the graph, such as its position attribute $v_{\texttt{pos}}$ is one of the selected samples (i.e. $\mathcal{V} \stackrel{\text{def}}{=} \{v \mid v_{\texttt{pos}} \in \{\xi_1,\ \ldots,\ \xi_m\}\}$). Then consider each sample ξ_i in S, and connect in the graph the two vertices whose positions are the closest to ξ_i (using d), if they have not been previously connected yet (see top line in fig. 1). This builds $\mathcal{E}$. Let us call this graph $\mathcal{G}$ the *auxiliary graph* since it will be used hereafter as a mean to compute $\mathcal{M}$-based geodesics.

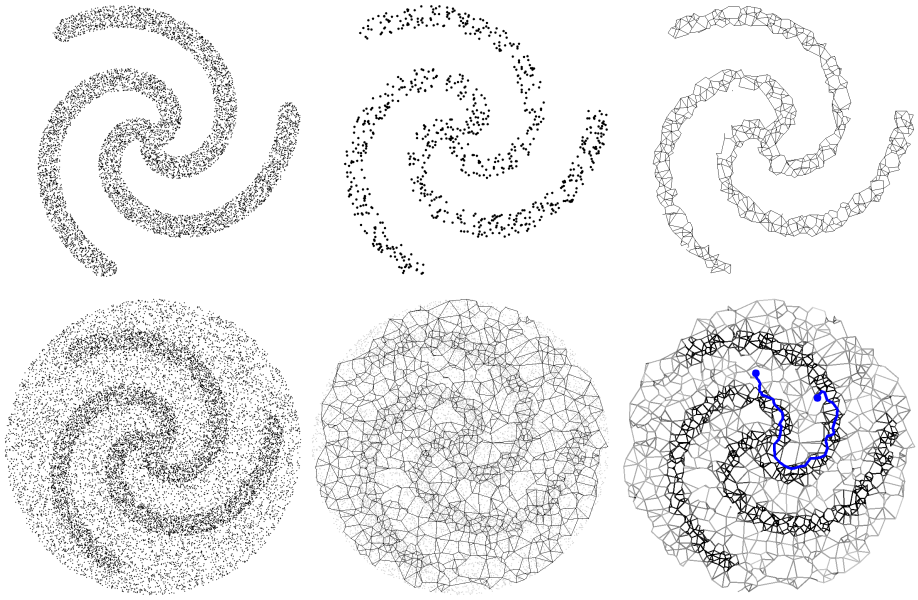

Fig. 1: Top line: The dataset S ($X = \mathbb{R}^2$), the m selected samples in S (i.e. $\mathcal{V}$), the edges $\mathcal{E}$ of $\mathcal{G}$ computed by CHL. Bottom line: A noisy dataset S, $|S| = 16046$ (the radius of the noisy disk is $\rho = 1.2$, the density of the noisy areas is 33% of the density of points in the manifold $\mathcal{M}$), the auxiliary graph $\mathcal{G}$, the edges densities of $\mathcal{G}$ (gray-scaled, black means $e_{\texttt{d}} = 1$, white would be 0).

The inner topology of the manifold $\mathcal{M}$ can be approximated by a topology induced by $\mathcal{G}$, as previously introduced in [4]. Indeed, the shortest path in $\mathcal{G}$ from one vertex position to another approximates the geodesics of $\mathcal{M}$ linking those two points. For points that are not exactly the vertices of $\mathcal{G}$, the geodesics

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

can be approximated as well. Let us detail here an improved version of the method introduced in [4]. The reader should refer to fig. 2-left while reading the definitions.

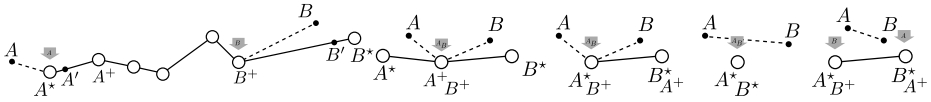

Fig. 2: The shortest path from $A^\star$ to $B^\star$ is in plain lines. The approximated graph-induced geodesics from A to B in the leftmost case is A, $A^\star_{\texttt{pos}}$, $A^+_{\texttt{pos}}$, …, $B^+_{\texttt{pos}}$, B. The two rightmost cases are the specific ones.

Let us consider any two points $(A, B) \in X^2$. Let $A^\star \overset{\text{def}}{=} \underset{v\in\mathcal{V}}{\operatorname{argmin}}\, \mathrm{d}\big(A,\, v_{\texttt{pos}}\big)$ be the closest vertex to A and $B^\star$ the closest vertex to B. Let us consider the cost $e_{\texttt{cost}}$, $e \in \mathcal{E}$ of an edge as its length $|e|$, i.e the distance d between the two positions of its extremities. This cost will be reconsidered later in section 3. Using that cost, the shortest path from $A^\star$ to $B^\star$ can be computed in $\mathcal{G}$ (we use A-star, the shortest path is represented in plain lines in fig. 2 examples). Let us denote by A^+ the vertex in that path following $A^\star$ and B^+ the vertex preceding $B^\star$. The *graph-induced geodesics* from A to B connects A and B to respective vertices on the path, either $A^\star$ or A^+ for A, either $B^\star$ or B^+ for B. We detail hereafter how to make that decision. The selected vertex is materialized by a gray arrow in fig. 2, one for A and another for B. The approximated geodesics is then the multiline starting from A, joining the shortest path by a segment to the selected vertex for A, following the path until the selected vertex for B and finally joining B with a segment (see fig. 2-left). Let us denote by $\mathrm{d}_{\mathcal{M}}(A,\, B)$ the length of that geodesics. It is the sum of the lengths (computed with d) of the segments composing the multiline. $\mathrm{d}_{\mathcal{M}}(A,\, B)$ approximates the distance separating A from B in the topology of $\mathcal{M}$.

In previous work [4], A and B were systematically connected to the path via $A^\star$ and $B^\star$, but in fig. 2-left, it can be seen that this is not a suitable choice for B. The improvement presented here consists in computing $A' \overset{\text{def}}{=} \mathrm{mv}\big(A^\star_{\texttt{pos}},\, A^+_{\texttt{pos}},\, \lambda\big)$ for a small positive λ. The basic interpolation is $\mathrm{mv}(x,\, y,\, \lambda) \overset{\text{def}}{=} (1-\lambda)x + \lambda y$. If $\mathrm{d}(A,\, A') < \mathrm{d}\big(A,\, A^\star_{\texttt{pos}}\big)$, we can estimate that a point moving from $A^\star_{\texttt{pos}}$ to $A^+_{\texttt{pos}}$ starts by approaching A before moving away from it. Otherwise, it starts directly by moving away from it. So in the first case, A^+ should be chosen instead of $A^\star$. The same is done for B. On fig. 2-left, this is why A joins the path at $A^\star$ (since $\mathrm{d}(A,\, A') > \mathrm{d}\big(A,\, A^\star_{\texttt{pos}}\big)$) while B joins the path at B^+ (since $\mathrm{d}(B,\, B') < \mathrm{d}\big(B,\, B^\star_{\texttt{pos}}\big)$).

Last, let us mention two specific cases (on the right in fig. 2). First, if $A^\star = B^\star$, then the graph-induced geodesics goes directly from A to B. The same stands if both A^+ and B^+ are chosen in a path made of two vertices.

A majority of vector quantization algorithms rely on the same learning rule, making a weight ω get closer to a sample ξ with the learning rate α. This

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

weight updating rule is expressed as $\omega \leftarrow (1-\alpha)\omega + \alpha\xi$. This corresponds to an interpolation, i.e. a walk, along the Euclidian geodesics from ω to ξ over a distance of $\alpha \mathrm{d}(\omega,\ \xi)$. Having $\mathcal{G}$ at our disposal for approximated geodesics in $\mathcal{M}$, we replace the learning rule by moving ω toward ξ along the approximated $\mathcal{G}$-induced geodesics defined previously. The final position of ω is obtained by moving ω along the geodesics over a distance of $\alpha \mathrm{d}_{\mathcal{M}}(\omega,\ \xi)$.

Self-organizing maps (SOMs) can be adapted with a graph-induced learning rule, as was done in [4], extending the capabilities of 1D-SOMs to approximate solutions of the Euclidian Travelling Salesperson Problem (TSP) to the non-Euclidian case. The present adaptation of the algorithm reduces by a factor 5 the number of vertices and edges for the same result (not shown here).

3 Graph-induced k-means

In this paper, let us focus on the online version of k-means [5]. Indeed, as opposed to the reference Linde-Buzo-Gray algorithm [6], it does not require the computation of a barycenter, which cannot be directly graph-induced as we did so far. The online version of k-means is recalled here (see algorithm 1). Here, beforehand, we have to set up $\mathcal{G}$. The winner-take-all selection (line 5) in algorithm 1) is made with the $\mathrm{d}_{\mathcal{M}}$ distance, and the learning step (line 6) is replaced by the walk along the graph-induced geodesics presented previously in section 2.

Algorithm 1 k-means online

1: // A dataset S is provided, $\alpha \in]0,\ 1]$.
2: Sample k prototypes $\{\omega_1,\ \ldots,\ \omega_k\}$ from S
3: **while true do**
4: Sample ξ from S.
5: $\kappa^\star \leftarrow \underset{\kappa\in\{1,\ \ldots,\ k\}}{\operatorname{argmin}}\ \mathrm{d}_{\mathcal{M}}(\omega_\kappa,\ \xi)$ // We use d for $\mathrm{d}_{\mathcal{M}}$ in the Euclidian case.
6: $\mathrm{mv}(\omega_{\kappa^\star},\ \xi,\ \alpha)$ // mv is $\omega_{\kappa^\star} \leftarrow (1-\alpha)\omega_{\kappa^\star} + \alpha\xi$ for Euclidian geodesics.
7: **end while**

Let us consider a noisy distribution, for the sake of handling robustness (see fig. 1-bottom-left). The computation of $\mathcal{G}$ leads to the graph depicted in fig. 1-bottom-middle. Indeed, as the vertices of $\mathcal{G}$ are randomly chosen, they can be samples at the noisy areas, even if the probability to be chosen in these areas is lower. Thus the edges of $\mathcal{G}$ cover the whole noise support, which is here the region of X that encloses $\mathcal{M}$. The consequence is that the graph-induced topology will represent that enclosing region, rather than $\mathcal{M}$ (see fig. 1-bottom-middle). To cope with noise and handle that drawback, we propose to weight the edges $\mathcal{E}$ of $\mathcal{G}$ (see fig. 1-bottom-right) and to consider these weights for the cost of the shortest path computations involved by the graph-induced learning rule.

The cost $e_{\texttt{cost}}$ of an edge e is redefined as follows. First, we count the number of samples $e_\#$ in a region surrounding the edge: $e_\# \overset{\text{def}}{=} |\{\xi \in S \mid \mathrm{d}(\xi,\ e) < \epsilon\}|$

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

($\mathrm{d}(\xi,\ e)$ is the distance from the sample to the segment represented by the edge). Let us normalize this count to get the edge density $e_{\mathtt{d}} \overset{\text{def}}{=} \mu e_{\#}/|e|$. The μ factor is such as a 1-length edge in an area with dense sample distribution as a density of 1.

The trick is then to compute the shortest paths from edges costs that depend on the edges' densities. Low-density edges have a very expensive cost, so the shortest paths consist in reaching the higher density areas as fast as possible, i.e. escaping from noisy areas to reach the closest area in $\mathcal{M}$, and then walk inside $\mathcal{M}$ according to the geodesics of its inner topology (see the blue line linking the two blue dots in fig. 1-bottom-right). We define for shortest path computation $e_{\mathtt{cost}} \overset{\text{def}}{=} \sigma(e_{\mathtt{d}})|e|$, where σ is a piecewise linear function. We use here $\sigma(0) = 1000,\ \sigma(0.4) = 900,\ \sigma(0.6) = 2,\ \sigma(1) = 1$.

For each learning rule application, once shortest paths are found with this cost, the interpolation is computed as detailed in section 2, i.e. considering strictly the edges' lengths defined by d.

The result of GIT k-means is depicted in fig. 3. Despite the high noise level, the weighting of the edges enables to place the $k = 6$ prototypes so that they clusterize the manifold $\mathcal{M}$. In fig. 3-left, two prototypes clusterize the noise, this is not an issue. What is relevant is that the prototypes placed inside $\mathcal{M}$ enable to clusterize its elements, according to the spiral topology. The Voronoï cells can be represented by coloring each position in X with the color of the closest prototype according to $\mathrm{d}_{\mathcal{M}}$. Both examples in fig. 3 show how the Voronoï tessellation is coherent with the topology of $\mathcal{M}$: frontiers separating Voronoï regions are not segments as with Euclidian Voronoï tessellation.

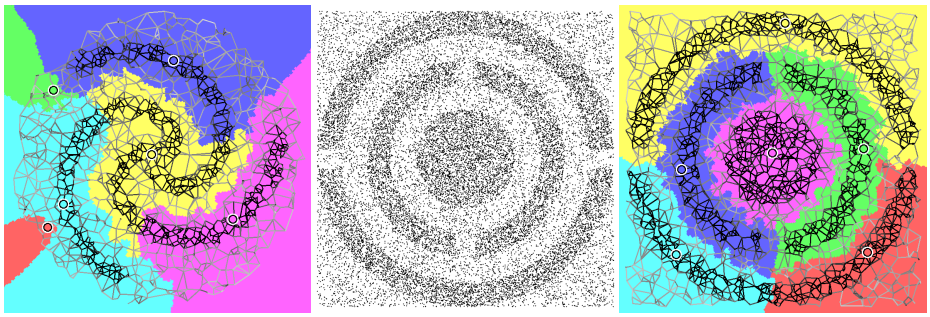

Fig. 3: Here, $k = 6$ prototypes are used for GIT k-means. The value of the learning rate α is successively taken in $\{0.2,\ 0.1,\ 0.05,\ 0.01\}$, 500 samples are presented for each value of α. Left is the result for the dataset introduced in fig. 1. The prototypes are the big circled color dots. In the background, each point is colored according to the color of its closest prototype, considering $\mathrm{d}_{\mathcal{M}}$. This reveals the Voronoï cells of that topology. Middle is another distribution, and right it the result obtained with the same parameters (which are $\gamma = 0.1,\ \lambda = 0.01,\ \mu = 1.43 \times 10^{-3},\ \epsilon = 0.05$).

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

The implementation of GIT k-means needs to be efficient since it massively invokes shortest path computations. Moreover, once $\mathcal{G}$ is set and once edges are weighted, from a mathematical point of view, learning rules are the same as the classical update, so all the vector quantization algorithms can be "GIT adapted" effortlessly. This requires a programming language that offers such generic programming features while preserving computational efficiency. To cope with both of these requirements, we propose an opensource `C++` library based on templates to implement this algorithm, and many others [7].

4 Discussion

In this paper, a method based on an auxiliary graph is presented, enabling to apply algorithms like SOMs and online k-means to non-Euclidian topologies. The geodesics corresponding to the inner topology of the manifold the data is taken from is approximated by the shortest paths in the auxiliary graph. The Euclidian learning rules are naturally adapted to follow these geodesics rather than the usual straight lines. The implementation only requires a distance d in the input space X, as well as an interpolation function between two elements of X, enabling to consider a variety of cases.

A specific edge weighting has also been introduced to handle noise. Edges of the auxiliary graph covering noisy regions are considered as much longer than others so that geodesics implicitly reach the closest region of the manifold before walking into it. Experiments on artificial data in a 2D space, for the sake of illustration, show promising results. As the mathematical rules used in this work are not restricted to 2D spaces, future work will address real data, as handwritten digits, represented as vectors. Indeed, for these image data, the Euclidian distance is not suitable, and walking along the manifold of digits could lead to improvement in clustering or mapping with self-organizing maps.

References

[1] András Király, Ágnes Vathy-Fogarassy, and János Abonyi. Geodesic distance based fuzzy c-medoid clustering – searching for central points in graphs and high dimensional data. *Fuzzy Sets and Systems*, 286:157–172, 2016.

[2] Thomas Martinetz and Klaus Schulten. Topology representing networks. *Neural Networks*, 7(3):507–522, 1994.

[3] Tuevo Kohonen. *Self-Organization and Associative Memory*, volume 8 of *Springer Series in Information Sciences*. Springer-Verlag, 1989.

[4] Hervé Frezza-Buet. Self-organizing maps in manifolds with complex topologies: An application to the planning of closed path for indoor UAV patrols. In *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2020.

[5] J. B. MacQueen. Some Methods for Classification and Analysis of MultiVariate Observations. In L. M. Le Cam and J. Neyman, editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.

[6] Yoseph Linde, Andres Buzo, and Robert M. Gray. Algorithm for Vector Quantization Design. *IEEE transactions on communications systems*, 28(1):84–95, 1980.

[7] Hervé Frezza-Buet. The vq3 library, 2022. https://github.com/HerveFrezza-Buet/vq3.