Vanilla Ice-Team: t-SNE & DBscan on structural complex datasets

Michele Avella, Elena Leonelli, Marika Sartore, and Filippo Ziliotto
(Dated: May 15, 2021)

In the era of big data, datasets are increasing in complexity and in dimensionality. For this reason, data mining techniques are widely spreading, including visualization and clustering methods. In this paper we present two unsupervised learning algorithms, one for data visualization *(t-SNE)* and the other for data clustering *(DBscan)*. These algorithms are indicated to extract patterns and information from high-dimensional data. We test their behavoiur on three different datasets: the first has a 5-dimensional knotted structure, while the others concerns binary data in 5 and 36 dimensions.

INTRODUCTION

In recent years managing and getting information from large datasets has become fundamental in many sectors. The process of extracting meaningful and interesting patterns and characteristics from complex datasets is named data mining, and includes different analysis methods.
An important data mining technique is visualization, which relies on capturing the qualitative structure of high-dimensional data, projecting them in lower dimensional space. In the last years has been introducted a nonlinear dimensionality reduction algorithm, t-Distributed Stochastic Neighbor Embedding (t-SNE) [1], which lends itself particularly to embedding high-dimensional datasets. For this reason t-SNE has become the standard for visualization in a wide range of applications, such as bioinformatics [2], cancer biology [3] and computer security [4].
Another data mining technique is clustering. Data clustering is an unsupervised learning technique based on the classification of data in different groups, according to points similarities. In this context, the DBscan algorithm [5] is one of the most versatile clustering algorithms used in many different fields, such as Web-based social network analysis [6] and temperature detection [7].
In this paper we analyze the behaviour of t-SNE and DBscan algorithms [8], focusing on their performances in high-dimensional and structural complex datasets.

t-SNE. The idea behind the t-SNE is to reduce the dimensionality preserving the local structure of data, so neighbour points in the original space will be neighbour in the latent space and distant points will be distant.
This is done by defining a $p_{i|j}$ for each point in the original space $x_i \in \mathbb{R}^p$:

$$p_{i|j} = \frac{\exp(-||x_i - x_j||^2/2\sigma_i)}{\sum_{k\neq i}\exp(-||x_i - x_k||^2/2\sigma_i)} \tag{1}$$

which can be interpreted as the likelihood that x_j is the neighbour of x_i ($p_{i|i} = 0$). σ_i are free bandwidth parameters that adapt to the density of the data: smaller values of σ_i are used in denser parts of the data space and vice versa. They can be derived by fixing the local entropy $H(p_i) = -\sum_j p_{j|i}\log_2 p_{j|i}$ and setting it equal to a constant $H(p_i) = \log_2\Sigma$ where Σ is the perplexity. We define $p_{ij} = (p_{i|j} + p_{j|i})/2N$.
T-SNE aims to find a map $y_i \in \mathbb{R}^{p'}$ (with $p > p'$) that maintains the similarity p_{ij} as maximum as possible; to do that we define a similar probability distribution q_{ij} on the latent space:

$$q_{ij} = \frac{(1 + ||y_i - y_j||^2)^{-1}}{\sum_{k\neq i}((1 + ||y_i - y_k||^2)^{-1})} \tag{2}$$

q_{ij} is a long-tail distribution that decrease more slowly and has a lower maximum than the Gaussian used before. In order to find y_i, we compare q_{ij} and p_{ij} using the Kullback–Leibler divergence:

$$D_{KL}(p|q) = \sum_{ij} p_{ij}\ln(\frac{p_{ij}}{q_{ij}}) \tag{3}$$

Since we want to keep the two distributions as closer as possible, we minimize D_{KL} using the gradient descent. The derivative is:

$$\partial_i D_{KL} = \sum_{i\neq j} 4p_{ij}q_{ij}Z_i(y_i - y_j) - \sum_{i\neq j} 4q_{ij}^2 Z_i(y_i - y_j) \tag{4}$$

with $Z_i = 1/\sum_{k\neq i}((1 + ||y_i - y_k||^2)^{-1})$. We call the fist term the *attractive term* and the second the *repulsive term*. When $p_{ij} > q_{ij}$ the *attractive term* is dominant, thus y_i and y_k try to stay closer as possible; on the contrary if $p_{ij} < q_{ij}$ the *repulsive term* is dominant, so y_i and y_k reject each other.
t-SNE can rotate data since D_{KL} is invariant under rotations on the latent space. In addition, the map y_i is stochastic because depends on the initial values of y_i. This algorithm is computational intensive $O(N^2)$ and can be improved to $O(N\ln N)$ by approximating Equation 4.

DBscan. The main idea behind this technique is that clusters are areas with high data density.
Considering data points $X = \{x_i\}_{i=1}^N$, we define x_c a *core-point* if:

$$\sharp\{x_i \in X \mid d(x_c, x_i) < \varepsilon\} < minPts \tag{5}$$

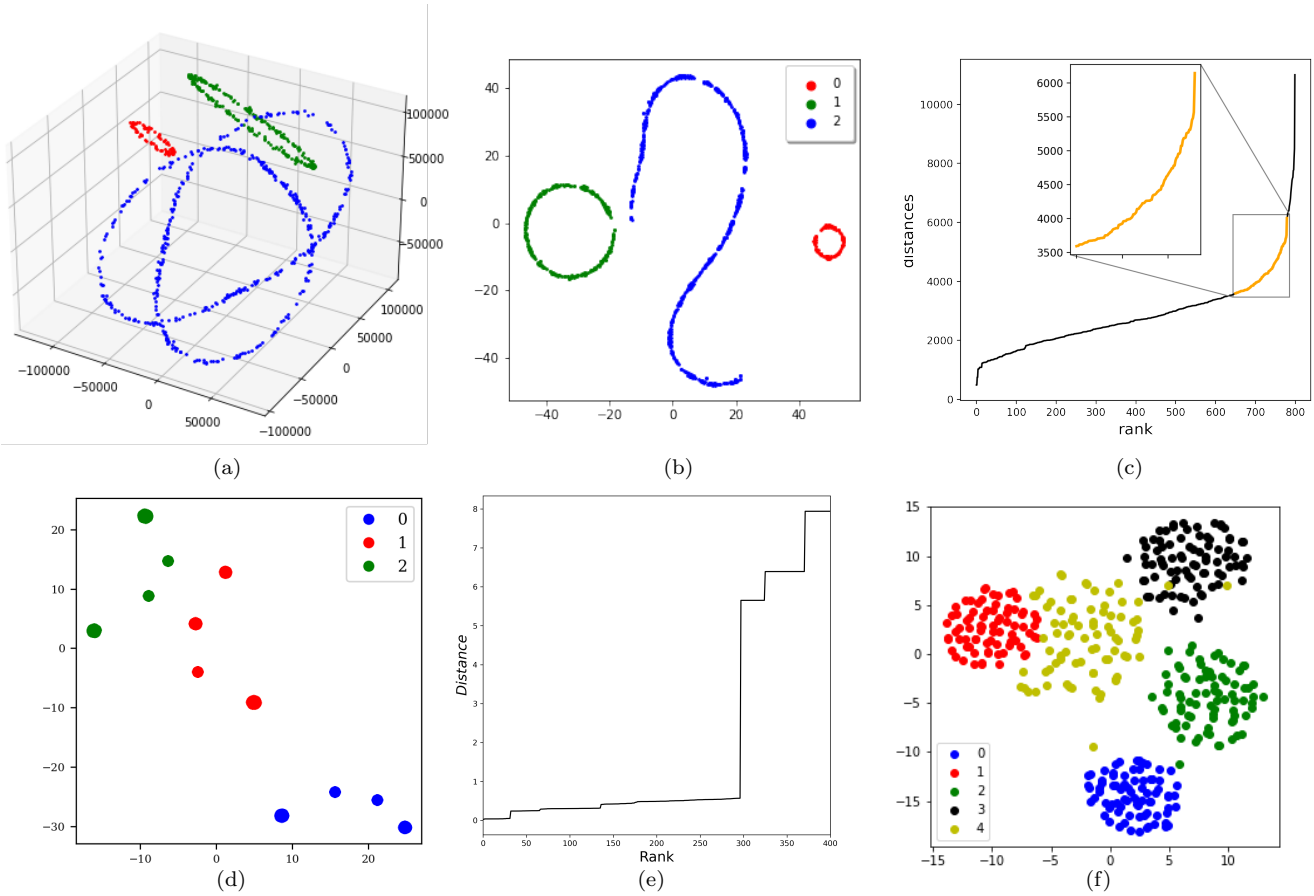

FIG. 1: (a) 5-dimensional non-binary dataset projected in the first three dimensions. The blue cluster is knotted with the green one. (b) t-SNE applied on the first dataset into 2 dimensions with perplexity 20. The algorithm separates very well the knotted structure seen in (a). (c) Distances between first nearest neighbours; the optimal choice for ε is a point in the range [3500,6300] (zoomed orange part), or a multiple of that value. (d) t-SNE applied to the binary 5-dimensional dataset. The algorithm is not able to recognize the different clusters very well in low dimensions for binary datasets. (e) Distances between 30th nearest neighbours for the 5-dimensional binary dataset. (f) t-SNE applied to the binary 36-dimensional dataset. Here we can visualize the clustered structure of the transformation, confirming the good choice of t-SNE in high dimensions.

where $d(\cdot,\cdot)$ is the euclidean distance, ε is a parameter and *minPts* is the minimum number of points to make a cluster. A point x_i is named *directly-reachable* from x_c if $d(x_c, x_i) < \varepsilon$, where x_c is a *core-point*. The point x_i is *reachable* from x_c if exist a set of points $\{x_k\}_{k=1}^{M}$ such that x_i is *directly-reachable* from x_M. x_{k+1} is *directly-reachable* from x_k $(\forall k)$ and x_1 is *directly-reachable* from x_c. At this point, starting from a *core-point* x_c we define cluster the set of all points that are *reachable* from x_c. Each cluster contains at least one *core-point*, and the non *core-point* of the clusters are the edges of the clusters. If a point is not *directly-reachable* from any *core-point*, it is considered as noise.

METHODS

To test the behaviour of t-SNE and DBscan algorithms, we perform an analysis on three different datasets, that can be found at [9]. The first dataset we consider has 5-dimensions, with 3 knotted clusters. The other two, 5-dimensional and 36-dimensional, contain binary features. These last are generated enforcing bit sequences to label the data. We create the binary 5-dimensional dataset with the purpose of studying the performances of the algorithms on binary data before scaling in higher dimensions.

We first apply t-SNE to the datasets to visualize them in the latent space. We set the perplexity parameter between 5 and 50, as suggested in literature [1]. The knowledge a priori of the number of clusters let us to find the optimal Σ value. For what concerns the number

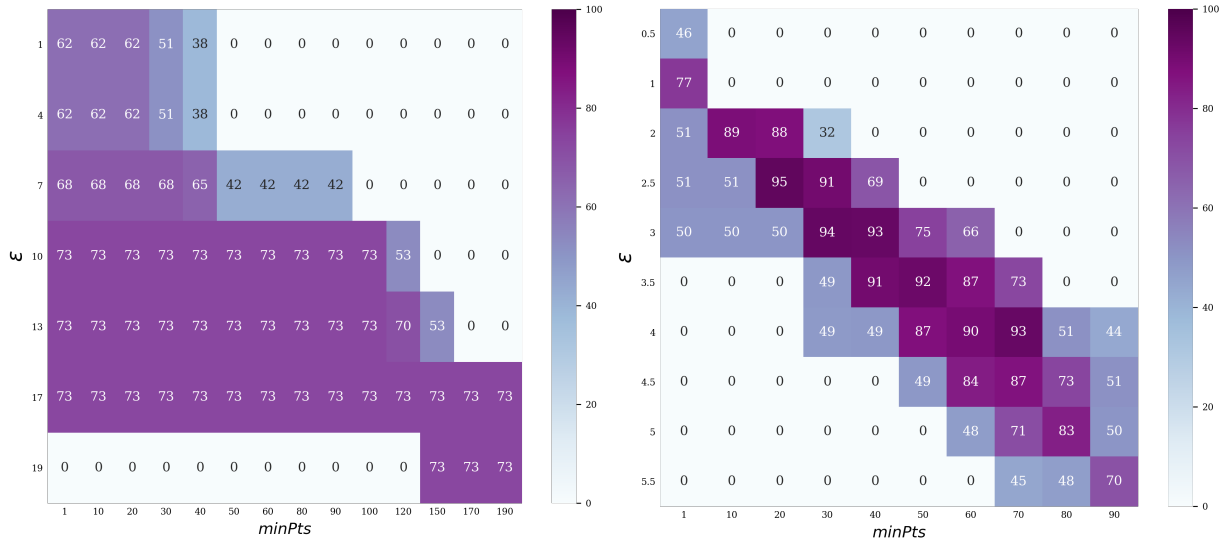

FIG. 2: (a) NMI heatmap for different choices of the parameter ε and $minPts$ of DBscan for the 5-dimensional binary dataset. We see how there is a wide range of optimal parameter performing reasonably well. (b) Same graph as (a), where here the NMI is calculated for the 36-dimensional binary dataset. Notice that increasing ε, $minPts$ decreases: the parameters have to be precisely tuned in order to find the best performances.

of iterations, the result of the algorithm, in most of the cases, converges around ~ 700 iterations.
To cluster the data we use DBscan algorithm. The ε parameter is estimated computing the distances to the first nearest neighbour. We plot the sorted distances and we selected the value where the tangent changes significantly. In all the datasets the optimal ε is a multiple of this point. We choose the $minPts$ by tuning different values; the best results are related to the structure of each dataset. The choice of the optimal parameters is based on the scores of Normalized Mutual Information (NMI [10]), which uses the true labels.
Eventually, we notice that it is useful to initialize DBscan using t-SNE when dealing with complex datasets.
For the 36-dimensional binary dataset we test some other clustering algorithms such as *KMeans*, *agglomerative clustering* and *spectral clustering* [11], in order to compare their performances with DBscan.

RESULTS

5-dimensional knotted dataset. The peculiarity of this dataset is that the clusters are knotted (see (Figure 1(a))), and we want to test how t-SNE and DBscan handle this kind of data. We perform t-SNE in a 2-dimensional latent space (Figure 1(b)), finding an optimal visualization with a perplexity $\Sigma = 20$.
DBscan is able to identify the clusters for this data with embedded manifolds, reaching an NMI value of 1 with a wide range of ε and $minPts$: $\varepsilon \in [17000, 26000]$, $minPts \in [1, 8]$. The optimal ε parameter is found between 3 and 5 times the value suggested by the first nearest neighbours plot, shown in Figure 1(c).

5-dimensional binary dataset. This dataset was generated with 3 labels: each label corresponds to a fixed sequence of the first three features, the last two features are random. We notice that the visualization with t-SNE is not efficient: the algorithm is not able to divide the data in the 3 clusters, as can be seen in Figure 1(d). This is due to the binary structure of the data and the low dimensionality, *e.g.* changing one over five bits changes radically the data position in the space. Looking at DBscan results, it does not properly work on the raw dataset, thus we initialize it with t-SNE.
In this case the performance of DBscan improves and reaches a NMI of 73%. Furthermore, this result is achieved with a wide range of ε and $minPts$, as shown in Figure 2(a). This is related to the data structure: the clusters created with t-SNE are very spread (see Figure 1(d)), and thus we can enlarge the radius ε without falling back into another cluster. In addition, there are many identical points in the dataset: in the latent space they overlap, and therefore $minPts$ can vary a lot without affecting the clustering. In this dataset the first nearest neighbour distance is not indicative of the optimal ε, because many points are overlapped. Instead, one can consider as more indicative a nearest neighbour

around the 30th (see Figure 1(e)).

36-dimensional binary dataset. To enlarge our analysis in higher dimensions, we studied a 36 dimensions binary dataset with five labels. It was generated in a similar way as before, with the sequences length that vary between 8 and 16 bits. t-SNE algorithm handles well the data structure, grouping together points with the same label Figure 1(f).
The most sparse cluster is related to the shortest fixed sequence (8 bits). Also in this case we fitted DBscan with the transformed t-SNE data. The results are shown in Figure 2(b): we can see that the highest NMI values (88%) are placed on the diagonal of the heatmap, because increasing the radius ε we expect to have more points inside the cluster. However these high values of NMI are limited over some specific pairs of (ε, $minPts$). These good performances of DBscan are attributable to the pre-processing done by t-SNE.
According to first nearest neighbors distances plot, we find the optimal ε values within 5 times the value suggested.
In order to have a more general approach we try other clustering algorithms: *KMeans*, *agglomerative clustering* and *spectral clustering*. All these algorithms perform well also on the raw dataset, after the indication of the number of clusters needed. To evaluate it, we perform a silhouette analysis [12]. It turns out that the best number of cluster is 5 with a silhouette coefficient value of 0.62, in perfect agreement with the number of labels of the dataset.
The results obtained are collected in Table I. We thus can notice that on this dataset DBscan is not the optimal choice for clustering.

NMI	Raw	t-SNE	Time*
DBScan	44	95	18
KMeans	99	96	177
Agglomerative	92	96	12
Spectral	98	94	205

TABLE I: Results for different algoritmhs tested for the high-dimensional binary dataset (for the raw data and the projected t-SNE data). **Computational time (in ms) of the algorithms is strongly dependent on the GPU and dataset used. The idea is to give a rough ratio estimate between the algorithms.*

CONCLUSIONS

t-SNE algorithm is successful in the visualization of a 5-dimensional embedded manifold dataset and a 36-dimensional binary dataset. On the contrary, it encounters difficulties in dealing with a 5-dimensional binary dataset because of its data structure (la spiegazione precisa deve stare nei results).
The performances of DBscan are good on a 5-dimensional embedded manifold dataset. With a binary dataset, to improve DBscan performances it is useful to initialize the data with t-SNE. This procedure leads to good clustering results in both a 5-dimensional and a 36-dimensional binary dataset. For future implementations, we suggest to analyze more deeply the relation between DBscan and binary datasets.

[1] L. Van Der Maaten and G. Hinton. *Visualizing Data using t-SNE.* **vol.9(86)**, 2579-2605 (2008).
[2] W. Li et al. *Application of t-SNE to human genetic data.* **vol.15(04)**, 1750017 (2017)
[3] W.M Abdelmoula et al. *Data-driven identification of prognostic tumor subpopulations using spatially mapped t-SNE of mass spectrometry imaging data.***vol.113(43)**, 12244-12249 (2016)
[4] Y. Hamid and Muthukumarasamy. *A t-SNE based non linear dimension reduction for network intrusion detection.* **vol.12**, (2019)
[5] E. Martin et al. *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.* 226–231 (1996).
[6] Y. M. ElBarawy et al. *Improving social network community detection using DBSCAN algorithm.* 1-6(2014).
[7] M. Çelik et al. *Anomaly detection in temperature data using DBSCAN algorithm.*, 91-95, (2011)
[8] P. Mehta et al. *A high-bias, low-variance introduction to Machine Learning for physicists.***vol.810**, (2018).
[9] `https://github.com/ZiliottoFilippoDev/Vanilla_Ice_Team_UL.git`
[10] P.A. Estevez et al. *Normalized Mutual Information Feature Selection.* **vol.20(2)**, 189-201 (2009).
[11] F. Pedregosa et al. *Scikit-learn: Machine Learning in Python.***vol.12**, 2825-2830 (2011)
[12] Peter J.Rousseeu. *Silhouettes: A graphical aid to the interpretation and validation of cluster analysis.* **vol.20**, (1986).