t-SNE
Epoch IIT Hyderabad

Ankit Saha
AI21BTECH11004

23 Jul 2022

1. Introduction

t-distributed stochastic neighbor embedding (t-SNE) is a non-linear dimensionality reduction algorithm. It converts the plot of higher-dimensional data into a lower-dimensional (usually 2D or 3D) plot while retaining most of the information and similarities between the data points. However, t-SNE does not preserve distances and densities. Hence, it is usually only used for visualization of high-dimensional data and not for clustering.

2. Mathematical Formulation

Suppose we are given m samples $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m$ where $\mathbf{x}_i \in \mathbb{R}^n \ \forall i \in [m]$

We are trying to map these inputs to d-dimensional outputs $\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_m$ where $\mathbf{y}_i \in \mathbb{R}^d \ \forall i \in [m], \ d \ll n$

Let $d(\mathbf{x}_i, \mathbf{x}_j)$ be a metric that denotes the distance between $\mathbf{x}_i$ and $\mathbf{x}_j$

In order to quantify the similarity between two data points $\mathbf{x}_i$ and $\mathbf{x}_j$, we can use the Gaussian distribution. The (unscaled) similarity of $\mathbf{x}_j$ with respect to $\mathbf{x}_i$ is given by

$$p_{j|i} = \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{d^2(\mathbf{x}_j, \mathbf{x}_i)}{2\sigma_i^2}\right), \qquad j \neq i \tag{2.1}$$

Since the densities of each cluster is not the same, the standard deviations of the distributions about each point are also different. This is why we normalize the above similarities so that the distribution of similarity scores are uniform for all points in the dataset.

$$\hat{p}_{j|i} = \frac{\frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{d^2(\mathbf{x}_j, \mathbf{x}_i)}{2\sigma_i^2}\right)}{\sum_{k \neq i} \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{d^2(\mathbf{x}_k, \mathbf{x}_i)}{2\sigma_i^2}\right)} = \frac{\exp\left(-\frac{d^2(\mathbf{x}_j, \mathbf{x}_i)}{2\sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(-\frac{d^2(\mathbf{x}_k, \mathbf{x}_i)}{2\sigma_i^2}\right)} \tag{2.2}$$

The similarity of $\mathbf{x}_j$ with respect to $\mathbf{x}_i$ is not necessarily equal to that of $\mathbf{x}_i$ with respect to $\mathbf{x}_j$. Thus, we choose the similarity of $\mathbf{x}_i$ and $\mathbf{x}_j$ to be the average of the two.

$$\hat{p}_{ij} = \begin{cases} \dfrac{\hat{p}_{j|i} + \hat{p}_{i|j}}{2m} & i \neq j \\ 0 & i = j \end{cases} \tag{2.3}$$

Note that

$$\begin{aligned}
\sum_{i,j} \hat{p}_{ij} &= \frac{1}{2m}\left(\sum_{i=1}^{m}\sum_{j=1}^{m} \hat{p}_{j|i} + \sum_{j=1}^{m}\sum_{i=1}^{m} \hat{p}_{i|j}\right) && (2.4)\\
&= \frac{1}{2m}\left(\sum_{i=1}^{m} 1 + \sum_{j=1}^{m} 1\right) && (2.5)\\
&= \frac{1}{2m}(m + m) && (2.6)\\
&= 1 && (2.7)
\end{aligned}$$

Now that we have a measure for similarity between two data points, our goal is to find a d-dimensional map whose distribution of similarity scores is similar to the distribution of similarity scores in the original n-dimensional space. The similarity scores in the d-dimensional space are calculated using the t-distribution instead of the Gaussian distribution, hence the name t-distributed stochastic neighbor embedding.

We first construct a similarity matrix containing the pairwise similarity scores for each pair of points in the original dataset. We then randomly project the input data points into d-dimensional space and construct a similarity matrix for this as well. For each point in this lower-dimensional space, we move it such that the row corresponding to it in the new similarity matrix becomes like the corresponding row in the original similarity matrix. Finally, we are left with the appropriate clusters that visualize the original dataset.

Alternately, the problem is equivalent to minimizing the Kullback-Leibler divergence of the two similarity scores distributions. The Kullback-Leibler divergence is a measure of how similar or different two distributions are.

3. Questions

i) What does linear and non-linear mean in the context of dimensionality reduction?
ii) What does similarity between two data points mean?
iii) Why is the Gaussian distribution used to model the similarity scores in the higher-dimensional space?
iv) Why is the t-distribution used to model the similarity in the lower-dimensional space?
v) Explain why t-SNE is not used for clustering.

4. Answers

i) Dimensionality reduction involves finding a map from a higher-dimensional space to a lower-dimensional space. A map $T : V \to W$ is said to be linear if $\forall\, \mathbf{x}, \mathbf{y} \in V\ \forall\, \alpha, \beta \in \mathbb{R}$,

$$T(\alpha\mathbf{x} + \beta\mathbf{y}) = \alpha T(\mathbf{x}) + \beta T(\mathbf{y}) \tag{4.1}$$

and non-linear otherwise

ii) Similarity between two data points can be interpreted in many ways. One common way is to consider points close to each other as similar. This is the basis of clustering.
In the context of t-SNE, the similarity of $\mathbf{x}_j$ with respect to $\mathbf{x}_i$ is defined as the conditional probability $p_{j|i}$ that $\mathbf{x}_i$ would pick $\mathbf{x}_j$ as its neighbour if the likelihoods of picking neighbours were normally distributed centered at $\mathbf{x}_i$

iii) The Gaussian distribution is used because the probability density values of points far from the mean are very small and those for points closer to the mean are high. This is consistent with our definition of similarity, i.e. low similarity for farther points and high similarity for nearer points.

iv) The PDF of a t-distributed random variable looks very similar to that of a Gaussian distribution but has a lower peak and higher tails. The t-distribution is chosen for making the final plot so that the clusters do not all clump up in the middle. The higher tails ensure that the clusters are spread out as less similar pairs are penalized less heavily. This ultimately makes the visualization easier.

v) One of the reasons is that t-SNE does not preserve distances and densities in the original dataset. It only preserves the similarities. This can cause t-SNE to produce "fake" clusters, clusters that are visually pleasing to see but are not very meaningful.
Another reason is that it does not learn a specific function that maps the higher-dimensional data to lower-dimensional data. It only arranges them in a step-by-step manner. When a new data point is added to the dataset, we not be able to map this to our lower-dimensional space and thus, we will have to make all the computations again to accommodate this new point.