t-Distributed Stochastic Neighbourhood Embedding

Abstract

In this manuscript we present a concise introduction to the t-SNE technique which involves configuring probability distributions to provide a low-dimensional representation of some dataset. This in turn allows one to extract information regarding the structure and clustering of the data from the original higher-dimensional map.

1 Stochastic Neighbourhood Embedding

The SNE technique was first introduced by Hinton and Roweiss [HR02] and forms the basis for the t-SNE technique. It begins by considering each sample as a point in n dimensions, where n denotes the number of variables. Each point x_i is then considered individually and assumed to have a Gaussian distribution centered about itself. As a result, one may describe the similarity of any other point x_j to x_i through the conditional probability

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

where the denominator serves as a normalising constant. We also set $p_{i|i}$ to zero for convenience. One should observe that $p_{i|j} = p_{j|i}$ cannot be implied from the above so that $p_{j|i}$ cannot be assumed to be symmetric. We shall discuss how to determine the σ_i later.

A similar procedure is followed when considering the lower-dimensional counterparts y_i. Each point y_i is again considered individually and assumed to have a Gaussian distribution centered around itself. In this particular case however, we assume that the Gaussian has the same variance for each point y_i. This is primarily because calculating σ_i is a computationally intensive task. We need only perform it once for the high-dimensional probabilities above, but doing it for hundreds or thousand of iterations whilst determining the optimal set of low-dimensional points y_i is unfeasible. Consequently, we let $\sigma_i = \frac{1}{\sqrt{2}}$ for all i. This particular standard deviation is chosen out of convenience: choosing anything else will only result in a rescaled version of the

final map. Therefore, the similarity of a point y_j to y_i is modelled by

$$q_{j|i} = \frac{\exp\left(-\left\|y_i - y_j\right\|^2\right)}{\sum_{k \neq i} \exp\left(-\left\|y_i - y_k\right\|^2\right)}$$

where once again the denominator serves as a normalising constant and $q_{i|i} = 0$. Having defined a probability distribution for each point in both the high and low dimensions, we can now measure the mismatch between them using the Kullback-Leibler divergence. Moreover, one can define a natural cost function by summing over the mismatch between all probability distributions P_i and Q_i (centered about the respective datapoints x_i and y_i) to get

$$C = \sum_i KL\left(P_i \| Q_i\right) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}.$$

SNE then attempts to minimise the cost function above via gradient descent. The gradient at each point y_i can be calculated and is given by

$$\frac{\delta C}{\delta y_i} = 2 \sum_j \left(p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j}\right)\left(y_i - y_j\right).$$

Initially, we do not have a set of points y_i to calculate the corresponding $q_{j|i}$, so a random sample is first generated from a Gaussian centered about the origin with relatively small variance. Explicitly, the update is then given by

$$\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} - \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t)\left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$$

where $\mathcal{Y}^{(t)}$ is usually a $n \times 2$ vector of two-dimensional y_i points at iteration t. The factor η (usually set around 100) is called the learning rate and $\alpha(t)$ (usually varying between 0.5 and 1) is the momentum, which aids the algorithm with "getting out" of sub-optimal local minima since the cost function is not convex.

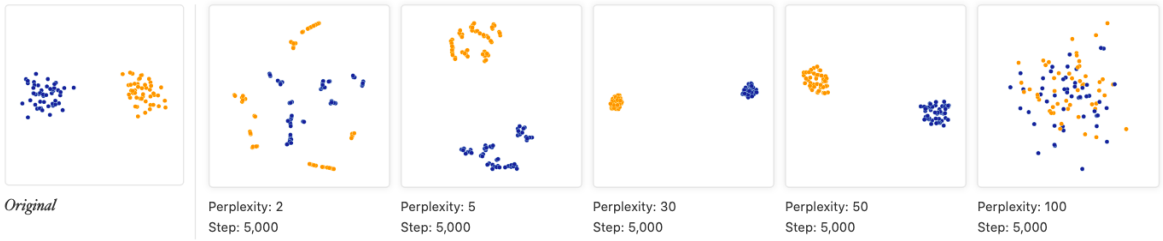
Before we consider other methods of improving the current SNE algorithm, we take a brief look at how the σ_i are determined for the high-dimensional probabilities $p_{j|i}$. The perplexity $Perp$ is global variable which is defined as

$$Perp\left(P_i\right) = 2^{H(P_i)}$$

where $H(P_i)$ is the Shannon entropy of the probability distribution centered about the point x_i, given by

$$H\left(P_i\right) = -\sum_j p_{j|i} \log_2 p_{j|i}.$$

When we say that the perplexity is a global variable we mean that $Perp(P_i) = Perp(P_j)$ for all i, j. The perplexity is usually set between 30 and 50, and by using the equations above along with the equation for $p_{j|i}$ one can can determine the corresponding σ_i (albeit not analytically). It is useful to notice that the perplexity increases monotonically with the standard deviations σ_i . Heuristically, the perplexity indicates the number of points within a cluster. The following plots below from [WVJ16] illustrate this well:



2 Symmetric SNE

When using the SNE algorithm it is most often the case that $p_{j|i} \neq p_{i|j}$ and $p_{j|i} \neq p_{i|j}$. In other words, the probability of x_j being the neighbour of x_i is not the same as the probability of x_i being the neighbour of x_j (and similarly for y_i, y_j). To remedy this one may consider the joint symmetric probability distributions defined by

$$p_{ij} = \frac{\exp\left(-\|x_i - x_j\|^2 / 2\sigma^2\right)}{\sum_{k \neq l} \exp\left(-\|x_k - x_l\|^2 / 2\sigma^2\right)}$$

and

$$q_{ij} = \frac{\exp\left(-\|y_i - y_j\|^2\right)}{\sum_{k \neq l} \exp\left(-\|y_k - y_l\|^2\right)}$$

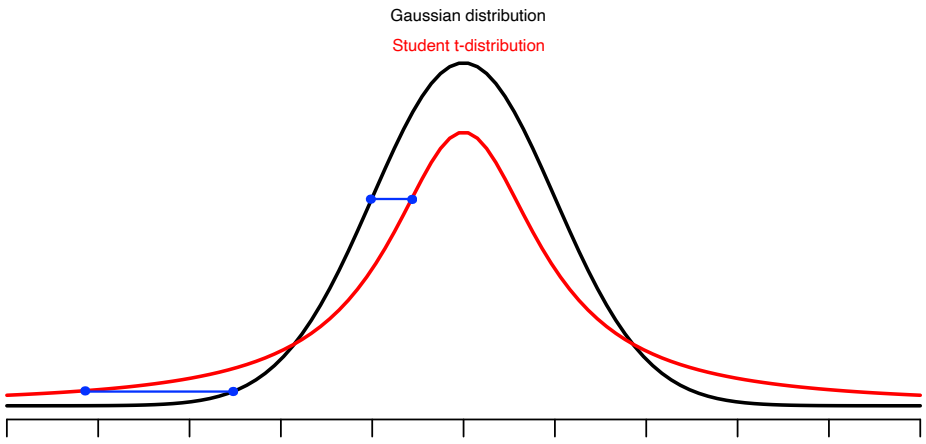
where the only difference is that we're now summing over all possible pairs in the denominator. However, a consequence of defining the higher-dimensional affinities in this fashion is that outliers are modelled quite badly. Indeed, consider an outlier x_i. Then $|x_i - x_j|$ is large for all x_j, which in turn provides a very small affinity p_{ij} for all points x_j. As a result, the cost function, by definition, is affected very little by the position of the outlier in the lower-dimensional map: varying the value of y_i and therefore q_{ij} does not produce a noticeable effect. To fix this, we let

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$$

which ensures that each point x_i makes a "significant" contribution since $\sum_j p_{ij} > \frac{1}{2n}$ for all i. This ensures that the positions of all the corresponding points in the lower-dimensional map are given some attention.

3 t-SNE

Although symmetric SNE manages to preserve the clustering, the low-dimensional representation of the data produced provides no evident distinguishment between these clusters: they all intersect each other to some degree. We'd like to "push" these clusters farther apart from each other. Using a Student t-distribution with one degree of freedom for the lower-dimensional distribution allows us to do this.



Recall that ideally, we hope to find a set of low-dimensional points y_k such that $p_{ij} = q_{ij}$ for all i, j, where we usually use Gaussian distributions for both p_{ij} and q_{ij}. However, consider instead switching to a Student t-distribution when modelling the q_{ij}. The fat tails of the t-distribution ensures that points which are "far apart" in the high-dimensional space are modelled by points even farther apart in the lower-dimensional space. Similarly, points which are "close" in the high-dimensional space will be modelled by points which are even closer in the lower-dimensional space. This forces the clusters in the lower-dimensional space to become more dispersed between themselves and therefore more distinguishable.

Mathematically the lower dimensional affinities will be given by

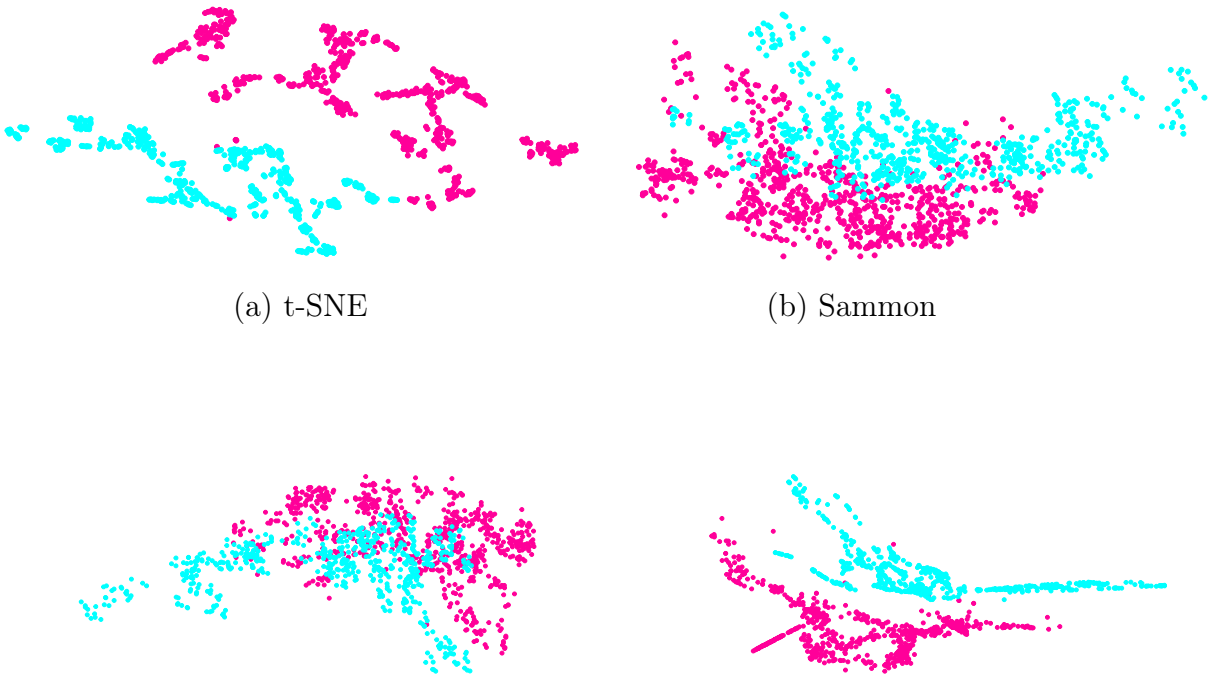
$$q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_{k \neq l} \left(1 + \|y_k - y_l\|^2\right)^{-1}}$$

and the corresponding gradient can be calculated to be

$$\frac{\delta C}{\delta y_i} = 4 \sum_j \left(p_{ij} - q_{ij}\right)\left(y_i - y_j\right)\left(1 + \|y_i - y_j\|^2\right)^{-1}$$

which is used with the aforementioned update to iterate through the lower-dimensional points.

Ultimately, t-SNE compares very favourably to the other data-reduction techniques with respect to data visualisation primarily due to its ability to create space between the various clusters. The visualisations below illustrate this well:



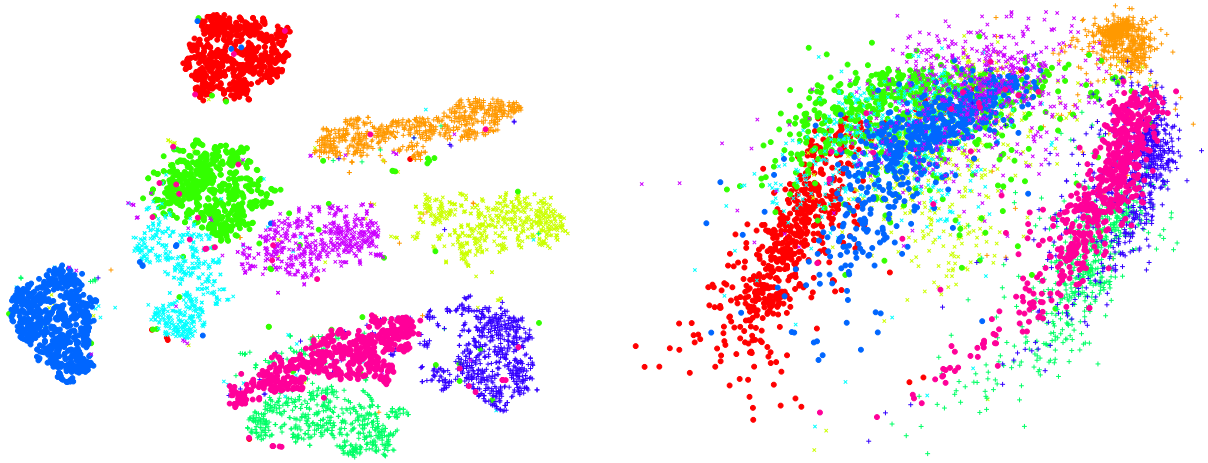
(a) t-SNE

(b) Sammon

(c) PCA

(d) Isomap

The plots above were produced using the Banknote Authentication dataset available from [Loh12]. In this particular instance, both t-SNE and Isomap perform particularly well, whereas the Sammon mapping and PCA suffer from what is known as the "crowding problem". However, for the larger MNIST dataset, Isomap is outperformed by t-SNE [VdMH08]:



(a) t-SNE (b) Isomap

4 Further Optimisations

1. The raw t-SNE algorithm is quadratic in both its computational and memory complexity. This constraint makes the algorithm unfeasible for larger datasets. To solve this problem, whilst also using all the available datapoints, one creates a neighbourhood graph and uses random walks between the points to calculate the high-dimensional affinities. A more detailed explanation of this technique can be found at [VdMH08].

2. To further aid with the memory complexity of the algorithm, one may apply Principal Component Analysis to initially reduce the dimension of the dataset. The dimensionality of the samples is usually reduced to around 30 before t-SNE is applied.

3. Using an adaptive learning rate, increasing the momentum over time and employing methods such as early compression and early exaggeration [VdMH08] to aid with the formation of clusters and create space can serve to make the algorithm

more effective in general.

References

[HR02] Geoffrey E Hinton and Sam Roweis. Stochastic neighbor embedding. *Advances in neural information processing systems*, 15, 2002.

[Loh12] Volker Lohweg. UCI machine learning repository, 2012. Available at `https://archive.ics.uci.edu/ml/datasets/banknote+authentication#`.

[VdMH08] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[WVJ16] Martin Wattenberg, Fernanda Viégas, and Ian Johnson. How to use t-sne effectively. *Distill*, 1(10):e2, 2016.