High Dimension Visualization: t-SNE

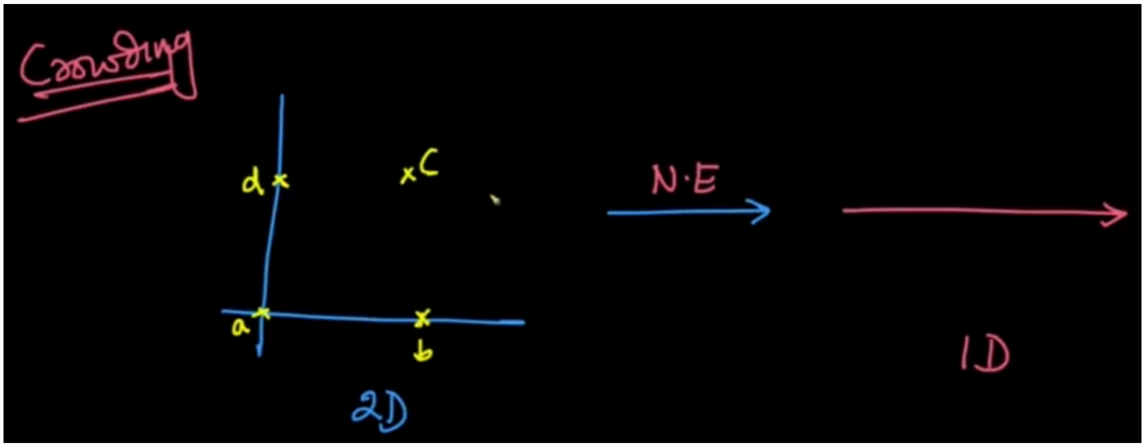
- t-SNE stands for **t-distributed Stochastic Neighborhood Embedding** which was presented by **Laurens van der Maaten** and **Geoffrey Hinton** in 2008.
- One of the limitations of PCA is that it does not preserve the neighborhood when points are projected from a higher dimension to a lower dimension.

- If one wants to project data from a higher dimension to a lower dimension, t-SNE will try to preserve the distances of the points that are close to each other.
- t-SNE tries to create an embedding that preserves the neighborhood using some probabilistic methods.

- Hence, the core idea behind t-SNE is;
    - When we go from d-dimensions to d'-dimensions where $d<d'$, the core idea behind t-SNE is to preserve the pairwise distance in a neighborhood as best as possible.
- But, there is a problem that t-SNE faces while preserving neighborhood information. It is known as **The Crowding Problem**.

Crowding Problem

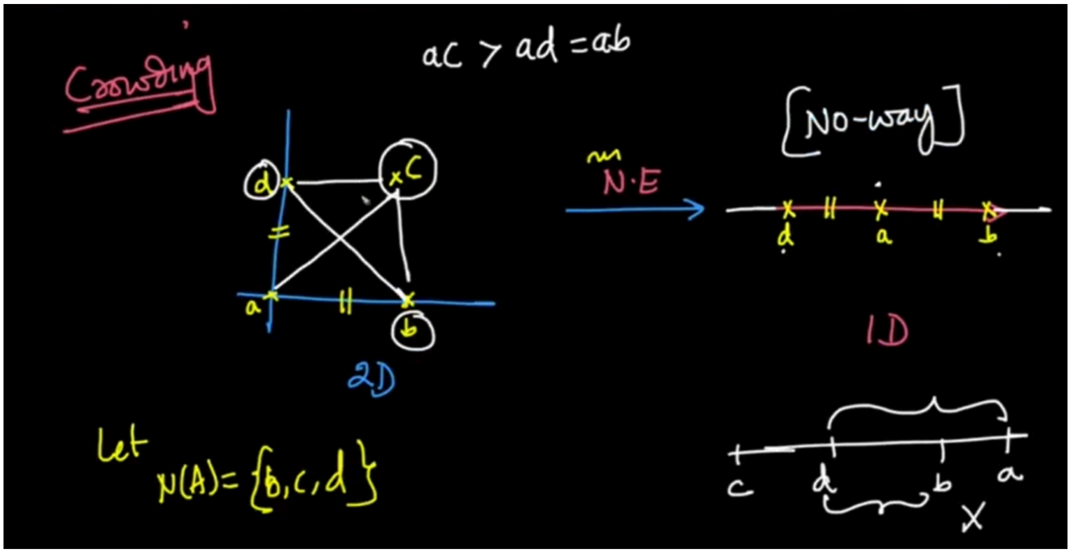
- Suppose we have 2D data and we want to project it in 1D data using any neighborhood embedding method.
- We have four data points in the shape of a square, where a is at the origin, b is on X-axis, and d is on Y-axis as shown in the diagram given below.



- Now, consider a case, when we choose the neighborhood of the point a that contains all the other points.

Let's try to project this data into 1D such that the pairwise distance is preserved

- We place point a on a 1D axis, point b on the right of point a, and point d on the left of point a. Here, the distance of both the points d and b to point a is the same.
- Now, if you try to project point c, it will be exactly projected at the coordinates of point a. Because, as a is equidistant from point b and d, so is the point c.



- This was just a simple case we saw for better understanding.
- In real-life data, there will be hundreds, probably thousands of points that will not be able to preserve pairwise distance when projected from a higher dimension to a lower dimension

Math for t-SNE

- Our objective is to project datapoints $xi \in R^d$ to y_i using t-SNE, where $y_i \in R^2$
- In t-SNE, we compute the pairwise similarities as probabilities.
- We compute P_{ij} for d-dimensions and Q_{ij} for $d^{'}$-dimensions where $d>d^{'}$
- P_{ij} is the probability that the points x_i and x_j are neighbors in d-dimensional space.

- The pairwise similarities in the low-dimensional map Q_{ij} are given by:

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

- The pairwise similarities in the high-dimensional space P_{ij} is:

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

- As you can see in the equation above, the numerator term in P_{ij} is nothing but a sort of normal distribution with a variance of σ.
    - The term $|x_i - x_j|^2$ computes the euclidean distance between x_i and x_j.
    - As x_i and x_j move farther and farther away, we give the lower probability that x_i and x_j are neighbors
- Now, as we are computing probabilities, probabilities across all points should be equal to 1
- So, the denominator terms is just a normalization factor to make sure that sum of all the probabilities is equal to 1
    - $\sum_i \sum_j P_{ij} = 1$

- This technique is known as SNE.

- Now, in $d^{'}$-dimensions space, every x_i and x_j would have corresponding y_i and y_j.
- So, again we define Q_{ij} with the same formulation as P_{ij}
- Hence, if x_i and x_j are similar, then P_{ij} would be higher and we want our y_i and y_j such a representation such that Q_{ij} is also high.
- Because of the crowding problem, we can never perfectly preserve the distance.

- So, in t-SNE, we try to preserve the probabilities when going from high dimension to low dimension space
- We compare probabilities P_{ij} and Q_{ij} with something known as KL-Divergence.

KL-Divergence

- It measures the dissimilarity between the distributions.
- So, the KL-Divergence between two distributions P and Q can be written as:

$$KL - div(P_{ij}, Q_{ij}) = \sum_i \sum_j [P_{ij} \ . \ log(\frac{P_{ij}}{Q_{ij}})]$$

- KL-divergence is also known as **relative entropy.**

Interpreting KL-Divergence

- If P_{ij} and Q_{ij} are the same, then KL-divergence will be equal to 0.
- If P_{ij} is very small and, P_{ij} and Q_{ij} are the same, then KL-divergence will have a small value.
    - Think of P_{ij} working as a weightage, because if P_{ij} is small we don't really care as points xi and xj will be far away from each other in d-dimension space.
- So, now our optimization problem would be to find all the y_is that minimize KL-divergence(P, Q)
- Lastly, since KL-divergence is a measure of dissimilarity, it is always greater than or equal to 0.

‘t’ in t-SNE

- For computing P_{ij}, we used gaussian like function.
- But, it was found that if we compute P_{ij} using t-distribution with 1 degree of freedom, the results were better.
- t-distributions with dof=1 have longer tails than gaussian distributions
- Gaussian distribution falls exponentially while t-distributions sort of inversely

- So, for our Q_{ij}s, if we start using t-distribution, two points can go farther away and still get pairwise distance preserved of sort
- Meaning, in t-SNE, we use Gaussian distribution for P_{ij}s and t-distribution for Q_{ij}s because of which if two points are far away in lower dimensional space, the probabilities will still remain the same
- Now, If we increase dof and keep increasing, it will behave like a gaussian distribution which will face the problem of crowding
- At, $dof=\infty$, it behaves very similar to a Gaussian distribution
- t-distribution with $dof=1$ is also known as Cauchy Distribution

Perplexity

- Perplexity is one of the most parameters that you might want to configure when using t-SNE.
- Perplexity can be interpreted as the effective number of neighbors whose distance we want to preserve
- Typically, we keep the value between [5,50]
- The optimization of t-SNE is very time taking as there are no single optima.
- Also, if you add a bunch of newer data points to the dataset, you won't get projections into lower dimensional space automatically.
- You would have to fit the t-SNE model again on the whole dataset again.

Use this blog to play around with t-SNE on different data distributions:
https://distill.pub/2016/misread-tsne/