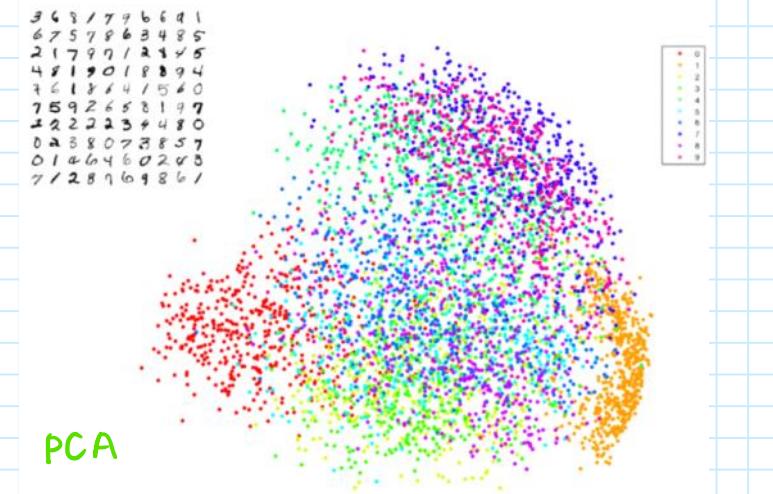


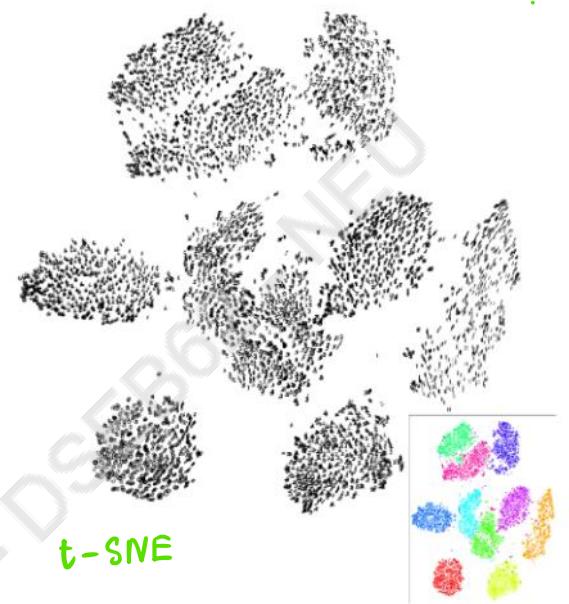
Lecture 2: t-SNE

① PCA review and introduction to t-SNE

- PCA is a dimensionality reduction algorithm that simply uses a linear projection matrix to project data. Problems arise when projected data is non-linearly separable, which happens usually in many cases.



Example : applying PCA and t-SNE on
MNIST dataset.



- Meanwhile, t-SNE (**t-distributed Stochastic Neighbor Embedding**) is also a dimensionality reduction algo that tries to preserves local structure, instead of finding a global structure for the whole dataset as in PCA.

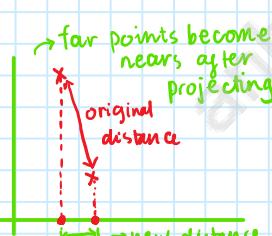
⇒ PCA and t-SNE comparison :

- Similarities:
 - Are dimensionality reduction algorithms.
 - Try to embed original data to lower dim subspace.

- Differences:

PCA

- Try to find global structure
- Project the whole dataset onto a lower dim subspace.
- Can lead to local inconsistency (far away point can become nearest neighbors)



t-SNE

- Try to preserve local structure.
- Embed neighbors of each points to lower dim space.
- Low dim neighborhood should be the same as original neighborhood.

② SNE basic idea

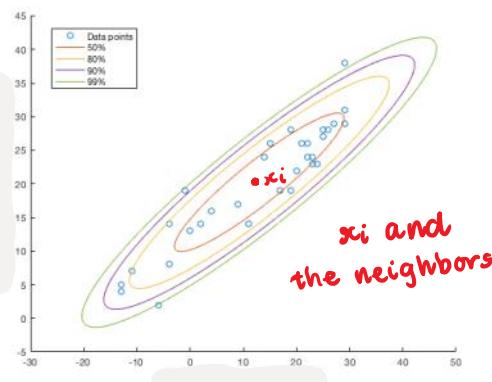
- Step 1 : In original dataset : "encode" high dim neighborhood information as a distribution. (for each point, if a neighbor points is near → high corresponding probability).

- Step 2: In lower dim space, find points such that their neighborhood distribution is similar to that of the original set.

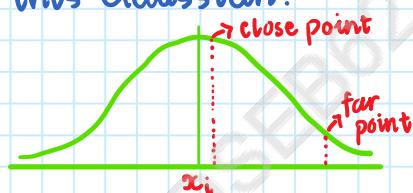
- ④ Intuition: Random walk between datapoints \rightarrow high probability to meet close point
- ⑤ How to measure the distance (or the similarity) between two distributions?
→ Using KL divergence.

③ The idea of neighborhood distribution.

- Consider a neighborhood around an input datapoint $x_i \in \mathbb{R}^d$.
- Imagine that we have a Gaussian distribution centered around x_i .



- Then the probability that x_i chooses some other datapoint x_j as its neighbor is in proportion with the density under this Gaussian.



- A point closer to x_i will be more likely than one further away.

④ Probabilistic view idea: distance between x_i and x_j will customize the probab.

The probability that point x_i chooses x_j as its neighbor:

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

↑ distance $x_i \rightarrow x_j$

and note that $p_{i|i} = 0$

↑ sum of distances of x_i to other points.

Final distribution over one pair is symmetrized:

$$p_{ij} = \frac{1}{2N} (p_{i|j} + p_{j|i})$$

↳ phân mẫu số của p_{ij}
và p_{ii} đều chia N components nên chia cho $2N$ để normalize.

Example:



→ By this step, we have found the joint distribution of all every pair of points in the dataset.

⑤ Distribution of low dimensional data.

- We consider low dim data as a parameter that we need to find:

Original data \rightarrow Low dim data

$$\begin{array}{ccc} x_1 & \rightarrow & y_1 \\ x_2 & \rightarrow & y_2 \\ \vdots & \vdots & \vdots \\ x_n & \rightarrow & y_n \\ P^D & \rightarrow & Q^d \end{array}$$

→ idea is to find the low dim data such that its neighborhood distr

Q is close to P .

- Now from high dim data x_1, \dots, x_n we already defined the distribution p_{ij}

C-goal: Find good embedding $y_1, \dots, y_n \in \mathbb{R}^d$ for some $d < D$ (normally $d=2$ or $d=3$)

- For point $y_1, y_2 \dots y_n$ we can define distribution Q similarly to P

(Notice that there is no σ_i in Qij formula and Q is not symmetric)

$$Q_{ij} = \frac{\exp\{-\|y^{(i)} - y^{(j)}\|\}}{\sum_k \sum_{l \neq k} \exp\{-\|y^{(l)} - y^{(k)}\|^2\}}$$

distance between x_i and x_j
→ distance of ALL pair of datapoints

có sự khác biệt trong mẫu số vì chiều xuống 2D hay 3D thì k/c giữa các điểm có sự thay đổi lớn nên chọn tổng k/c giữa các điểm làm mẫu số có ý nghĩa tổng quát tốt hơn.

- In the next step, we want to optimize Q close to P using KL divergence
The optimization problem turns to minimizing KL divergence (measure of distance between distributions P & Q).

⑥ KL Divergence

KL divergence of two distribution is given as :

$$KL(Q||P) = \sum_{ij} Q_{ij} \log\left(\frac{Q_{ij}}{P_{ij}}\right)$$

Properties: $\Rightarrow KL(Q||P) \geq 0$; " $=$ " $\Leftrightarrow Q = P$
 $\Rightarrow KL(Q||P)$ is a convex function

⑦ SNE Algorithm

- Finding $y_1, y_2 \dots y_n$ is in fact the work of building a model including parameters $y_1, y_2 \dots y_n$. Suppose the parameters follow distribution Q.
- Then loss of the model is $L(Q) = KL(P||Q)$. Loss function:

$$KL(P||Q) = \sum_{ij} P_{ij} \log\left(\frac{P_{ij}}{Q_{ij}}\right) = -\sum_{ij} P_{ij} \log(Q_{ij}) + \text{const}$$

so it is a constant to the model.

here, note that P is distribution of high dim data, which is known,

→ chú ý đây là đạo hàm của SNE

- Take partial derivative of $L(Q)$ w.r.t gradient vector: $\frac{\partial L}{\partial y^{(i)}} = \sum_j (P_{ij} - Q_{ij})(y^{(i)} - y^{(j)})$

The opt problem is solved by iterative algo such as GD...

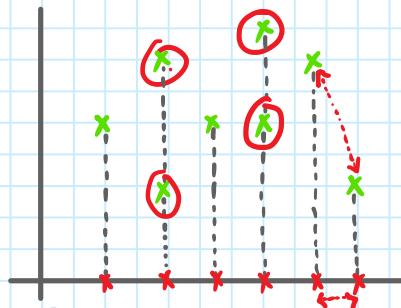
- The gradient vector of t-SNE algo is: $\frac{\partial L}{\partial y_i} = 4 \sum_j (P_{ij} - Q_{ij}) \frac{(1 + \|y_i - y_j\|_2^2)^{-1}}{(y_i - y_j)}$

⑧ Crowding problem.

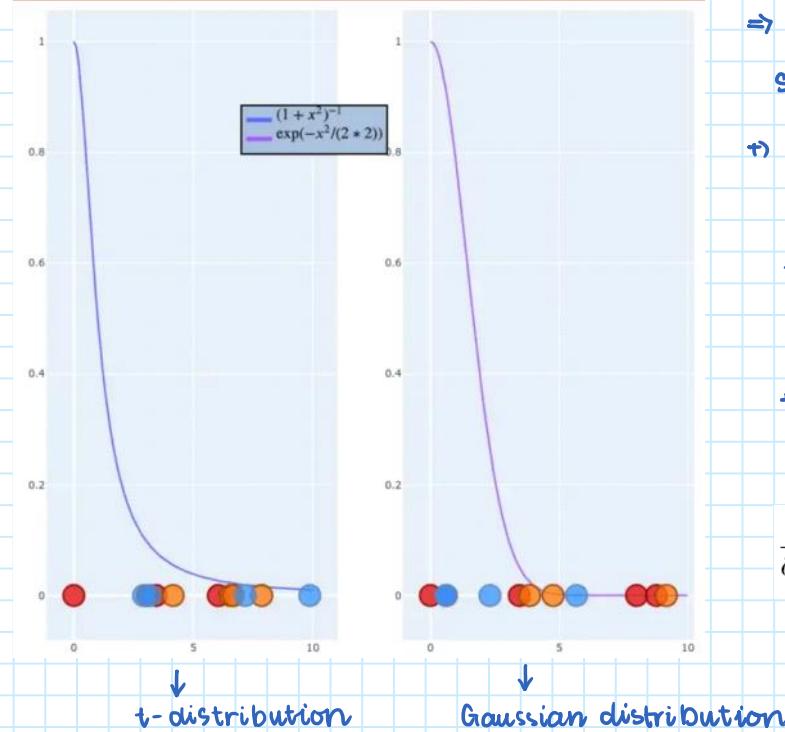
- In high dim we have more room, points can have different neighbors with various distances. In 2D a point can have few neighbors at distance one all far from each other.

Example : Cảm nhận, ta muốn chiếu dữ liệu từ 2D về 1D, có thể thấy tại 1D ko đủ chỗ nên nhiều điểm bị chồng lên nhau hoặc gần hơn thực tế. \Rightarrow vẫn dễ lỗi nhất của SNE.

\Rightarrow sự ra đời của t-SNE để giải quyết vấn đề này.



- t-SNE solution: với dữ liệu đặc điểm giảm chiều, ta sử dụng t-distribution thay vì Gaussian distribution để biểu diễn neighbor probability Q . (t-distribution cũng có dạng bell-shape như Gaussian distribution nhưng có tail dài hơn, nên các điểm ở xa cũng có thể tính được xác suất).



⇒ Có thể thấy với t-distribution, xác suất tiến về 0 chậm hơn nhiều so với Gaussian.
⇒ Công thức P_{ij} vẫn được giữ nguyên nhưng công thức Q_{ij} được redefined thành:

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

⇒ Như vậy, công thức tính gradient tương ứng với loss function mới là:

$$\frac{\partial L}{\partial y^{(i)}} = \sum_j (P_{ij} - Q_{ij})(y^{(i)} - y^{(j)}) (1 + \|y_i - y_j\|^2)^{-1}$$

③ Perplexity

- Có thể thấy trong công thức tính P_{ij} có sự xuất hiện của một tham số σ_i . Với Gaussian distribution, nếu σ_i càng lớn ⇒ bell shape càng thoái ra ⇒ xác suất các điểm gần nhau bé đi nhưng bù lại xác suất với các điểm xa bù lại có giá trị lớn hơn, ngược lại nếu σ_i bé thì distribution chỉ cover tốt những điểm gần.
- Vậy tựu chung lại σ_i thể hiện size của neighborhood quanh điểm x_i , nếu chọn σ_i quá thấp: chỉ biểu diễn probability của nearest neighbors.
⇒ σ_i quá cao: ⇒ cái bell shape thoái quá thành ra xác suất đến các điểm gần và xác suất đến các điểm xa không chênh lệch mấy (distribution tends to be uniform) ⇒ không thể hiện đúng khoảng cách xa gần.
- Vậy việc chọn σ_i tối ưu cho từng x_i là rất quan trọng, ảnh hưởng đến độ tốt của việc biểu diễn P_{ij} , xem size và cấu trúc của neighborhood cần preserve.
→ P_{ij} tốt thì sau đó mới có cơ sở để tìm Q_{ij} tốt.
- Perplexity là đại lượng dùng để đại diện cho chung các σ_i của các điểm, nó là số nguyên, thể hiện số lượng điểm được nằm trong neighborhood của x_i hay $P_{ij} \neq 0$, vì vậy điều chỉnh perplexity dễ hơn chọn giá trị cho σ . Theo thực nghiệm t-SNE có performance tốt nhất khi chọn perplexity trong khoảng 5 đến 50.
- Công thức tính perplexity như sau:

$$\text{perp}(P_{j|i}) = 2^{H(P_{j|i})}, H(P) = - \sum_i P_i \log(P_i) \text{ is the entropy}$$

- perplexity và ô lõi quan hệ thuận chiều:

⇒ Low perplexity = small σ.

⇒ High perplexity = large σ.

⇒ vì vậy perplexity cũng là tham số quan trọng.

- Minh họa ảnh hưởng của việc chọn perplexity:

