Manifold learning

t-SNE, UMAP, LLE, etc

Some slides courtesy Simon Carbonnelle, Université Catholique de Louvain

Readings

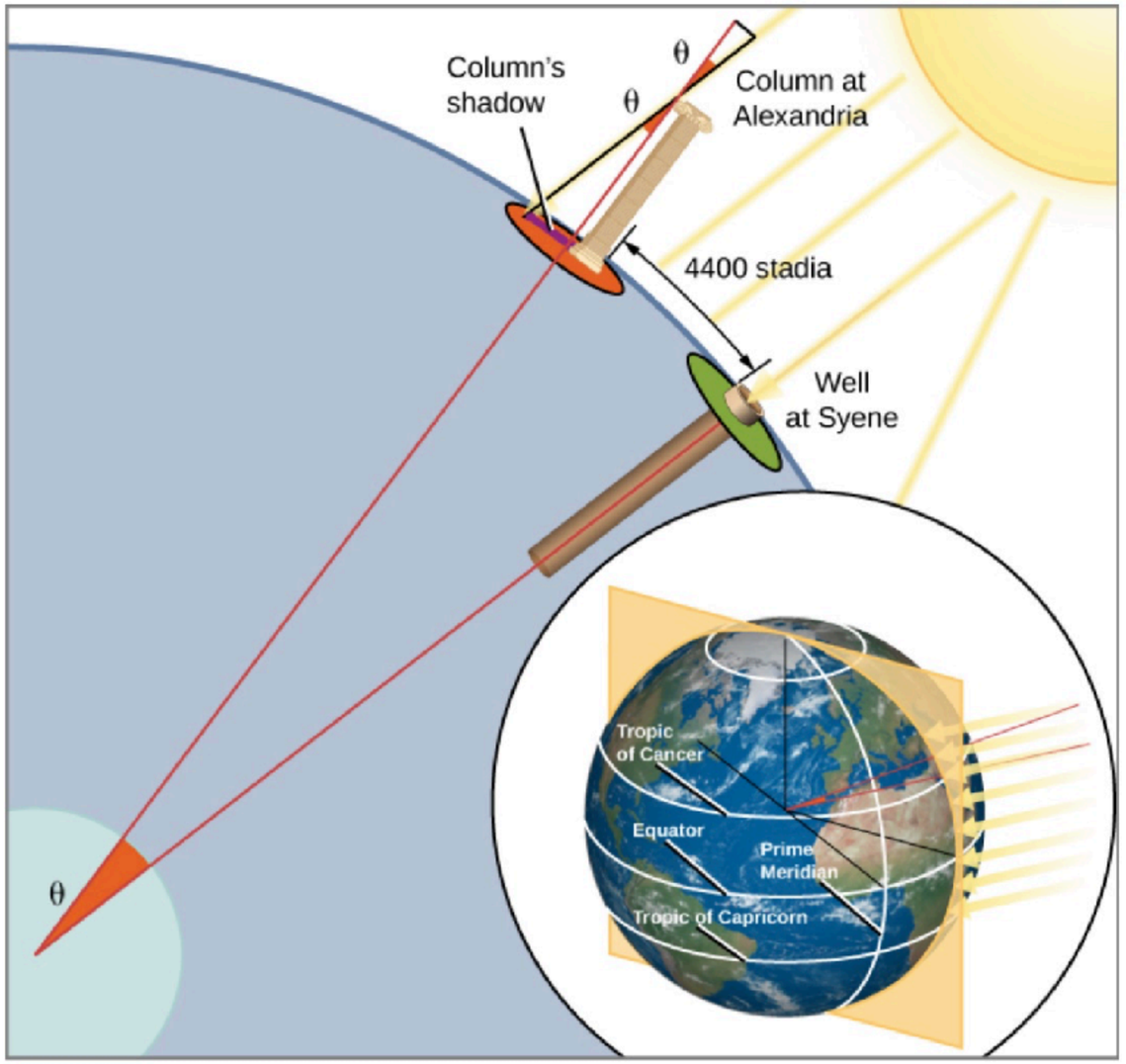
- https://jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf
- https://distill.pub/2016/misread-tsne/
- https://pair-code.github.io/understanding-umap/index.html
- https://umap-learn.readthedocs.io/en/latest/how_umap_works.html

Manifolds

- A manifold is a topological space that is locally Euclidean
- Consider the existence of FLAT EARTHERS
- On the small scales that we see daily the Earth does indeed look flat.
- Any object that is nearly "flat" on small scales is a manifold, and so manifolds constitute a generalization of objects we could live on :)

Manifolds
SIDE BAR

- A manifold is a topological space that is locally Euclidean
- Flat earth is not some ancient belief… the ancient Greeks and others were generally aware of the curvature of the Earth.
- Eratosthenes of Cyrene (3rd century BCE) was chief librarian at the Library of Alexandria & calculated the Earth's circumference with an error of ~2%



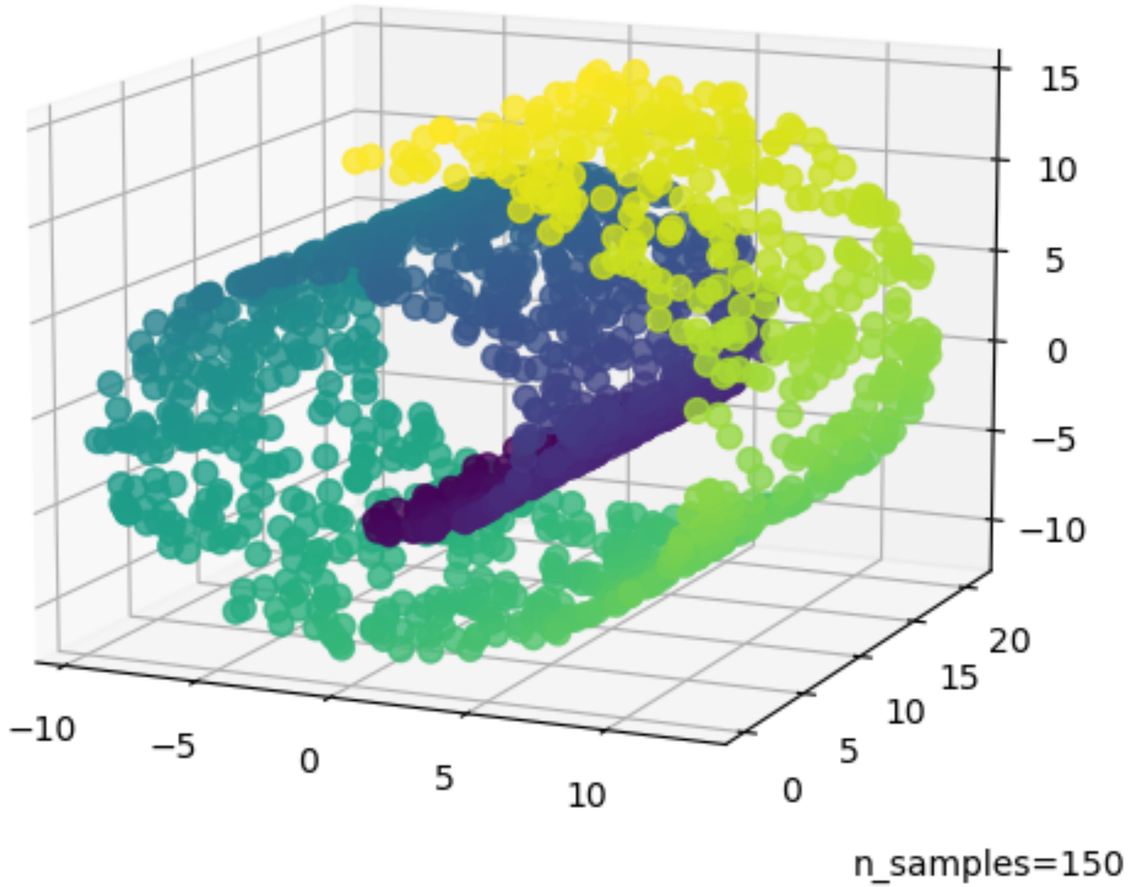
Topology
SIDE BAR

- One of the goals of topology is to find ways of distinguishing manifolds, e.g….
  - A circle is topologically the same as any closed loop
  - The surface of a coffee mug with a handle is topologically the same as the surface of the donut

Manifold learning

- Manifold learning is an approach to dimensionality reduction where the problem is only locally linear/Euclidean
- Algorithms for this task are based on the idea that the dimensionality of many data sets is only artificially high if you allow curved manifolds

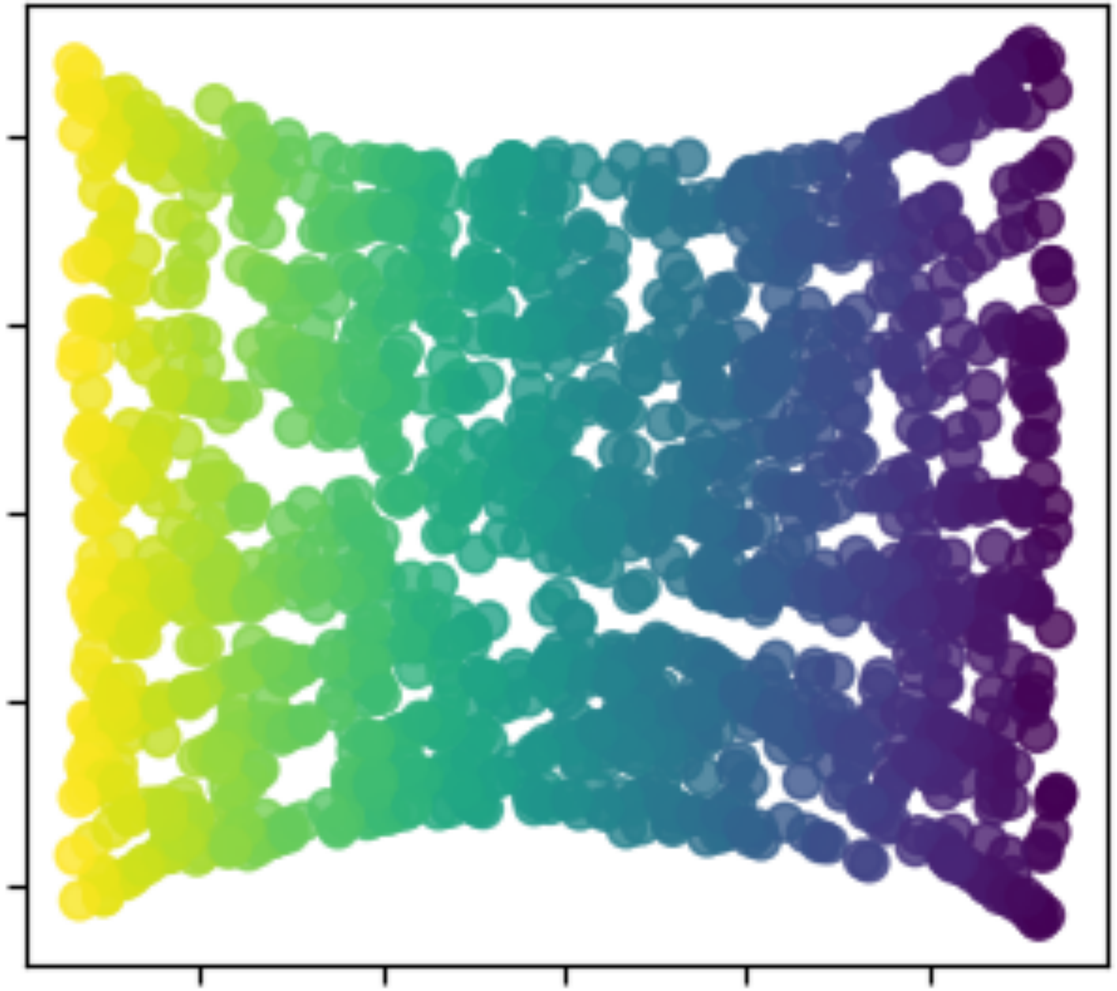
Swiss roll dataset



Manifold learning

- Manifold learning is an approach to dimensionality reduction where the problem is only locally linear/Euclidean
- Algorithms for this task are based on the idea that the dimensionality of many data sets is only artificially high if you allow curved manifolds

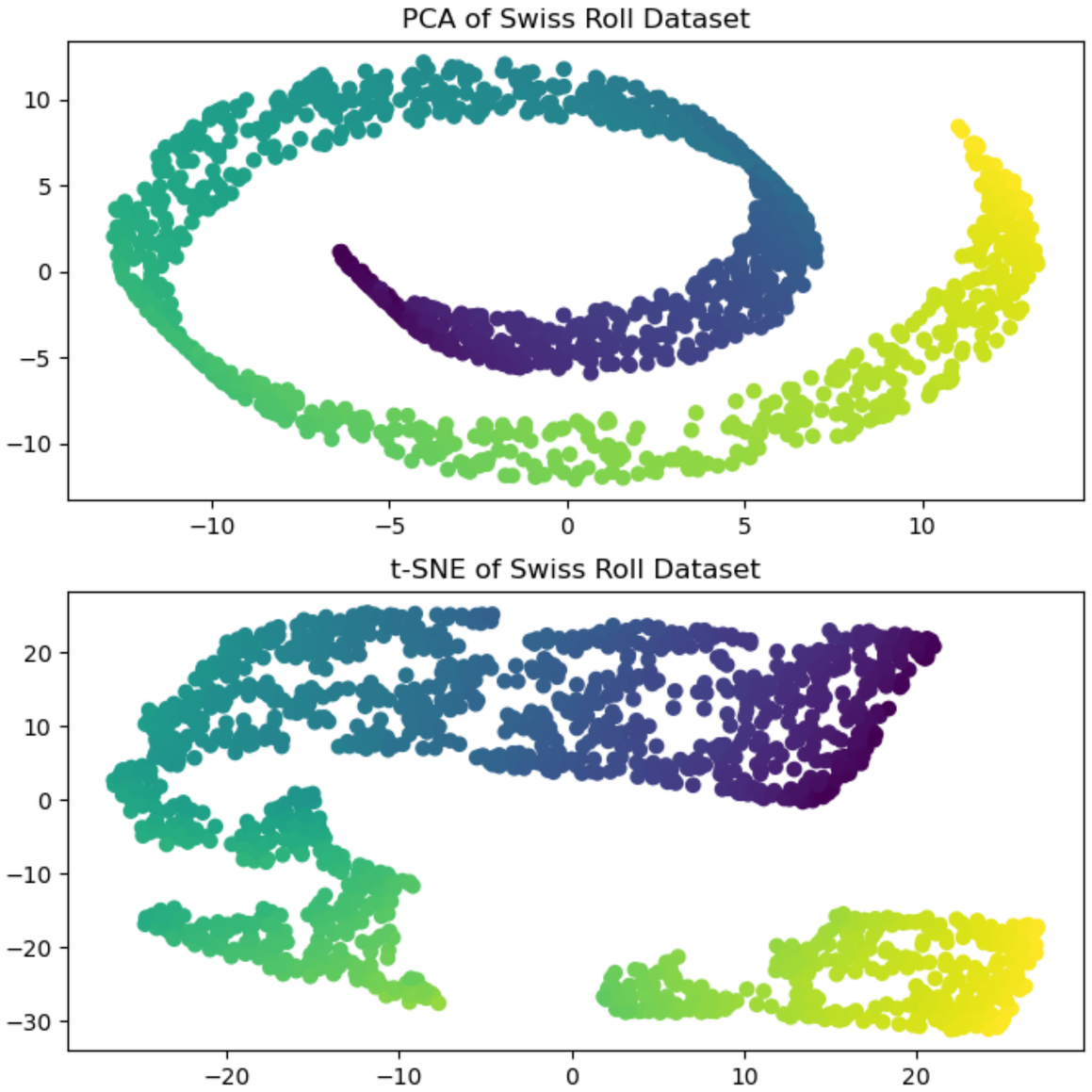
"Idealized" 2D manifold for Swiss Roll

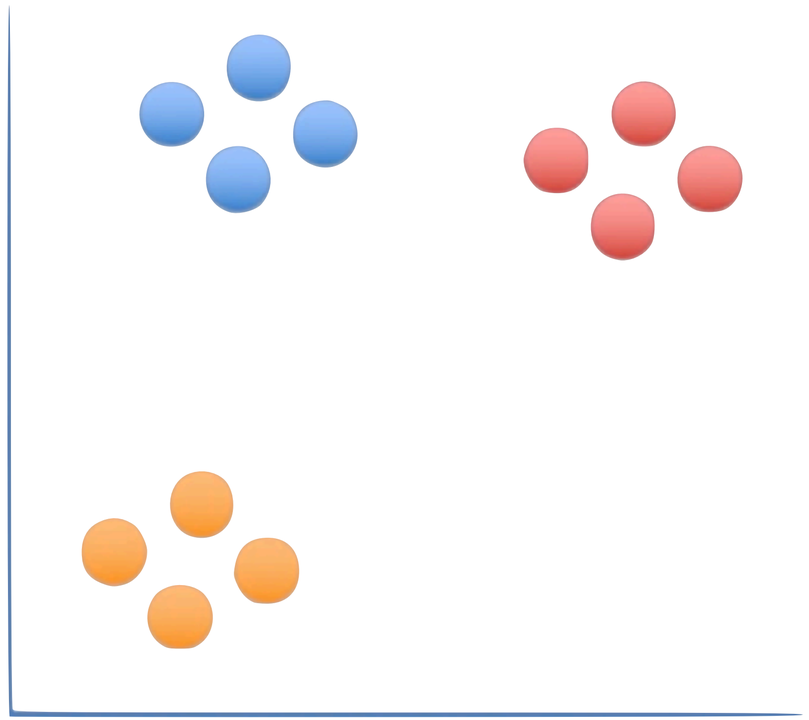


t-SNE

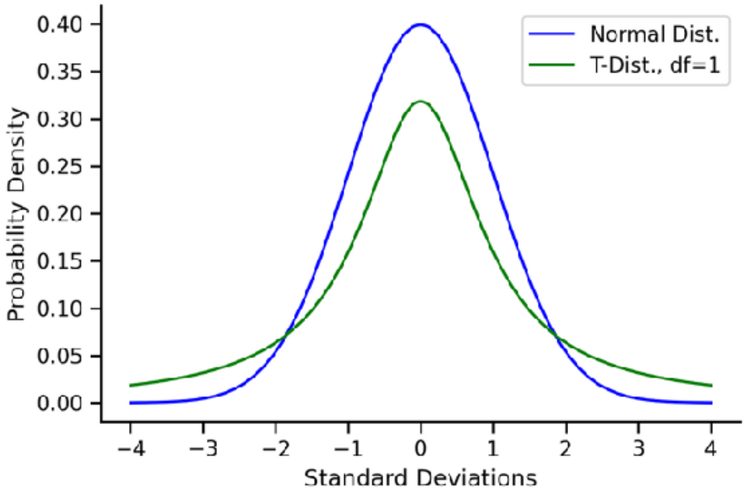
t-distributed Stochastic Neighbor Embedding

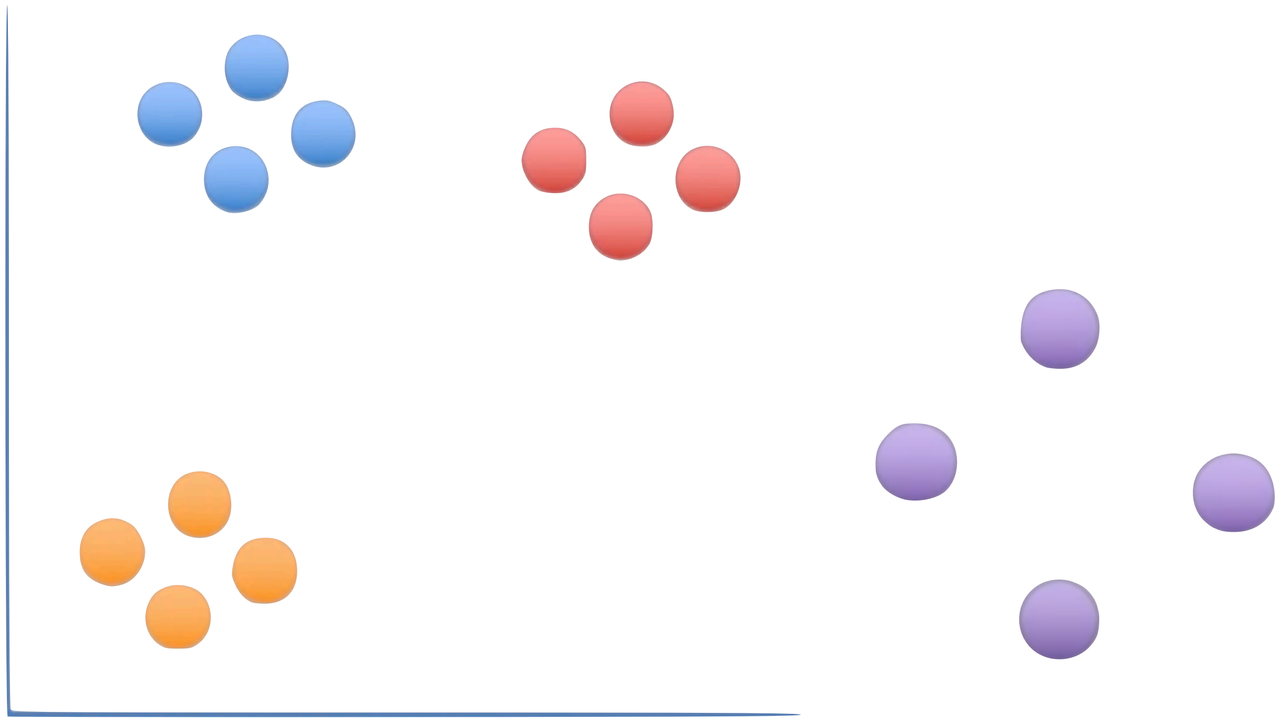
- PCA tries to find a global structure
  - Can lead to local inconsistencies in a subspace…far away points can become nearest neighbors
- t-SNE tries to preserve local structure
  - “For high-dimensional data that lies on or near a low-dimensional, non-linear manifold it is usually more important to keep the low-dimensional representations of very similar datapoints close together, which is typically not possible with a linear mapping”





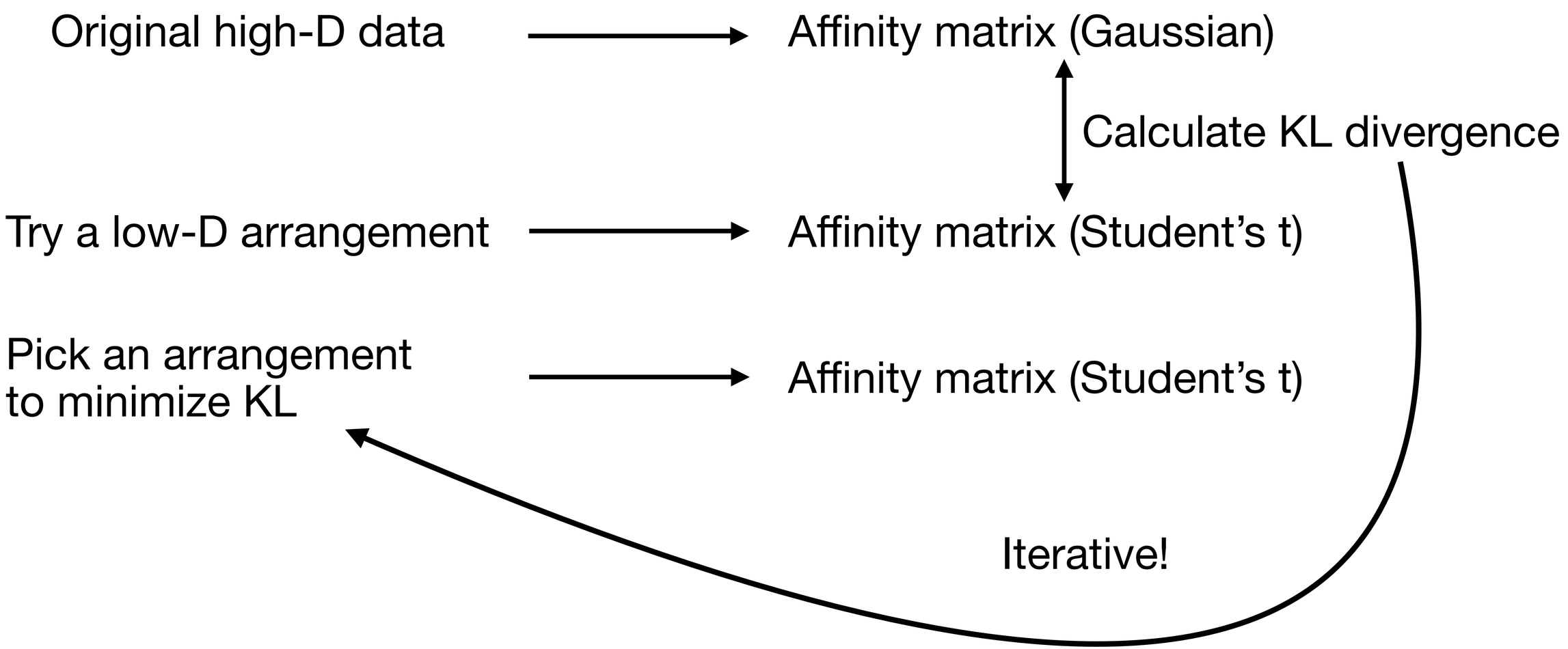
https://www.youtube.com/watch?v=NEaUSP4YerM





https://www.youtube.com/watch?v=NEaUSP4YerM

t-SNE concepts



Different approaches to Manifold Learning

- You can optimize
  - Distance preservation
  - Topology preservation
  - Information preservation

$$\mathcal{X} = \{x_1, x_2, ..., x_n \in \mathbb{R}^h\} \rightarrow \mathcal{Y} = \{y_1, y_2, ..., y_n \in \mathbb{R}^l\}$$

$$\min_{\mathcal{Y}} C(\mathcal{X}, \mathcal{Y})$$

- tSNE is designed to preserve distances, but in doing so tends to also preserve topology

https://lvdmaaten.github.io/publications/papers/JMLR_2014.pdf

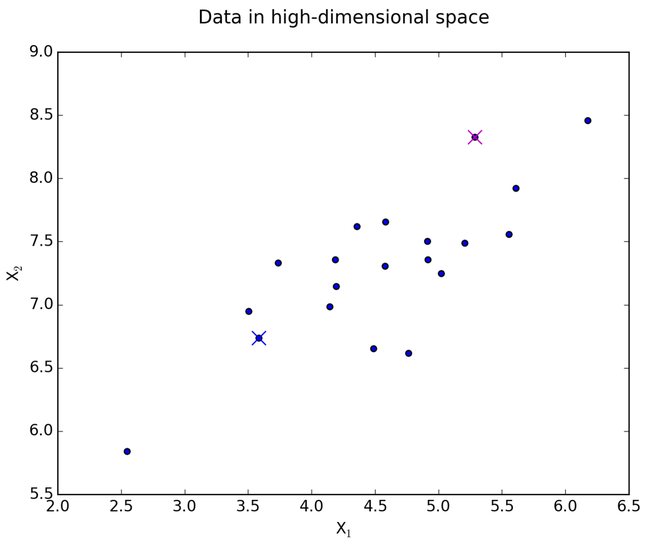
t-SNE algorithm

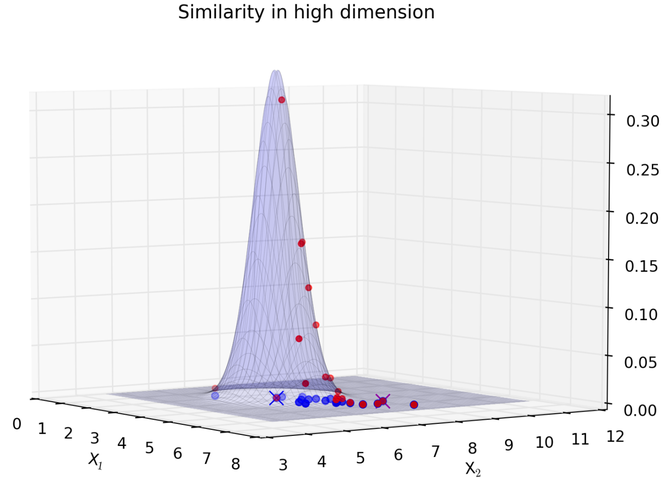
points in the embedding. Assume we are given a data set of (high-dimensional) input objects $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ and a function $d(\mathbf{x}_i, \mathbf{x}_j)$ that computes a distance between a pair of objects, *e.g.*, the Euclidean distance $d(\mathbf{x}_i, \mathbf{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\|$. Our aim is to learn an s-dimensional embedding in which each object is represented by a point, $\mathcal{E} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N\}$ with $\mathbf{y}_i \in \mathbb{R}^s$ (typical values for s are 2 or 3). To this end, t-SNE defines joint probabilities

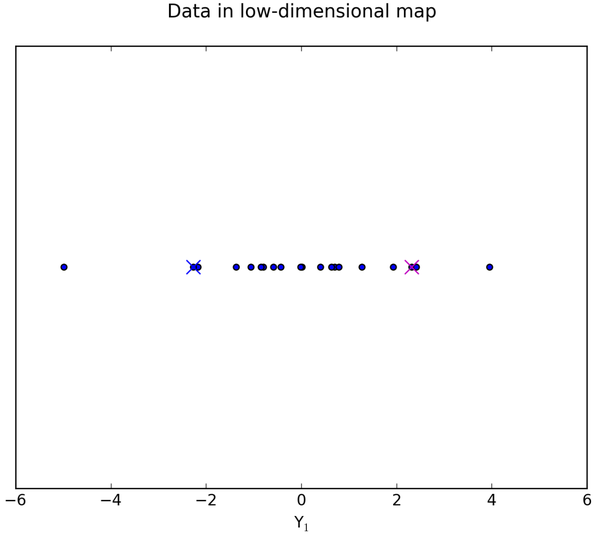
$$p_{j|i} = \frac{\exp(-d(\mathbf{x}_i, \mathbf{x}_j)^2/2\sigma_i^2)}{\sum_{k \neq i} \exp(-d(\mathbf{x}_i, \mathbf{x}_k)^2/2\sigma_i^2)}, \qquad p_{i|i} = 0$$

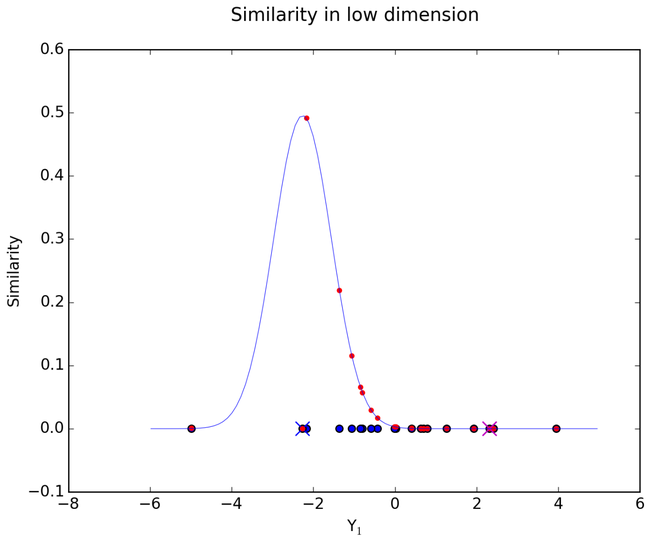
$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}.$$

Pair-wise similarities should stay the same.

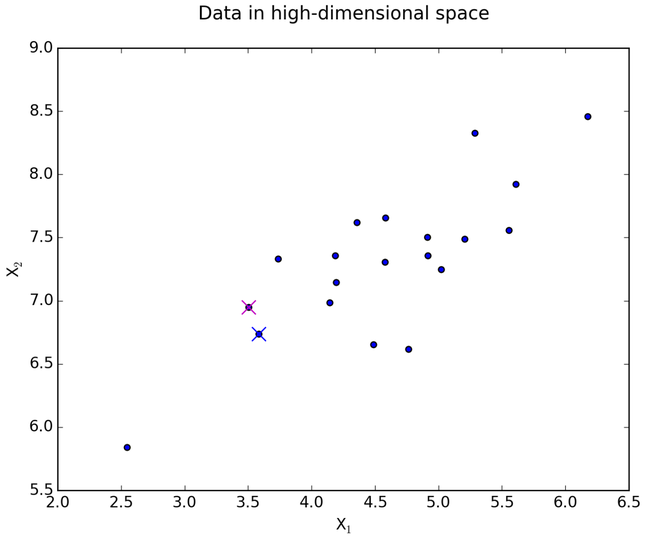


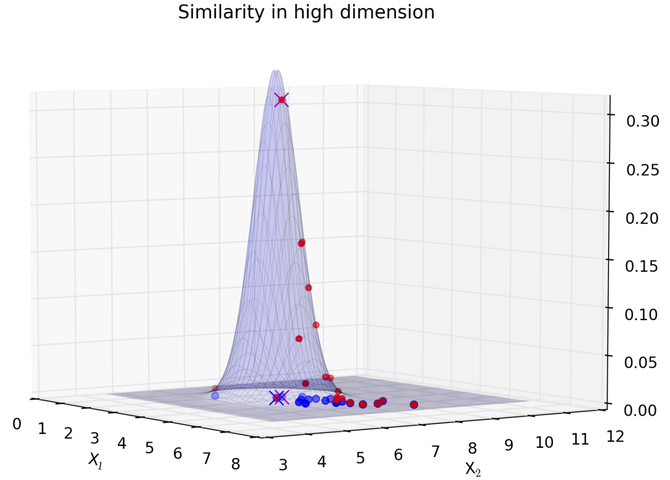


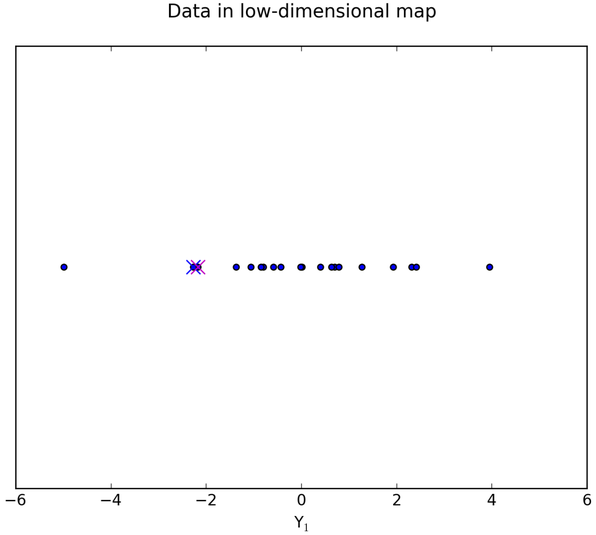


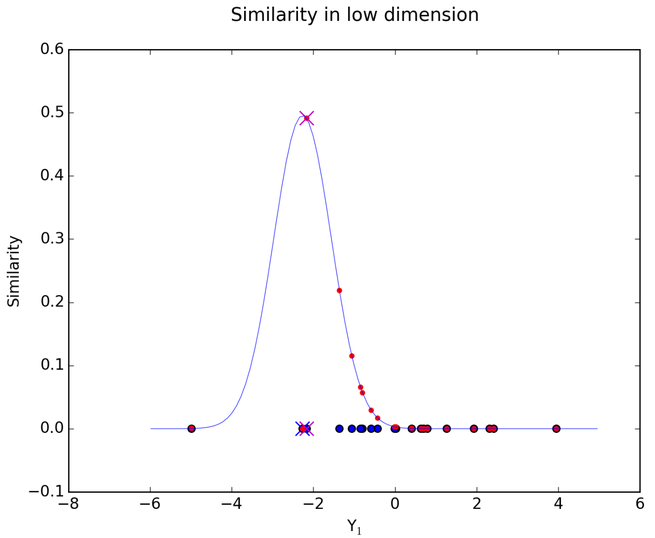


Pair-wise similarities should stay the same.









https://lvdmaaten.github.io/publications/papers/JMLR_2014.pdf

t-SNE algorithm

In the s-dimensional embedding $\mathcal{E}$, the similarities between two points $\mathbf{y}_i$ and $\mathbf{y}_j$ (*i.e.*, the low-dimensional models of $\mathbf{x}_i$ and $\mathbf{x}_j$) are measured using a normalized heavy-tailed kernel. Specifically, the embedding similarity q_{ij} between the two points $\mathbf{y}_i$ and $\mathbf{y}_j$ is computed as a normalized Student-t kernel with a single degree of freedom:

$$q_{ij} = \frac{(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}}{\sum_{k \neq l}(1 + \|\mathbf{y}_k - \mathbf{y}_l\|^2)^{-1}}, \qquad q_{ii} = 0. \tag{3}$$

The heavy tails of the normalized Student-t kernel allow dissimilar input objects $\mathbf{x}_i$ and $\mathbf{x}_j$ to be modeled by low-dimensional counterparts $\mathbf{y}_i$ and $\mathbf{y}_j$ that are too far apart. This is desirable because it creates more space to accurately model the small pairwise distances (*i.e.*, the local data structure) in the low-dimensional embedding.

The locations of the embedding points $\mathbf{y}_i$ are determined by minimizing the Kullback-Leibler divergence between the joint distributions P and Q:

$$C(\mathcal{E}) = KL(P||Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}. \tag{4}$$

Kullback-Leiber Divergence measures the faithfulness with wich $q_{j|i}$ models $p_{j|i}$.

- $P_i = \{p_{1|i}, p_{2|i}, ..., p_{n|i}\}$ and $Q_i = \{q_{1|i}, q_{2|i}, ..., q_{n|i}\}$ are the distributions on the neighbors of datapoint i.
- Kullback-Leiber Divergence (KL) compares two distributions.

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

- KL divergence is asymmetric
- KL divergence is always positive.
- We have our minimization problem: $\min_{\mathcal{Y}} C(\mathcal{X}, \mathcal{Y})$

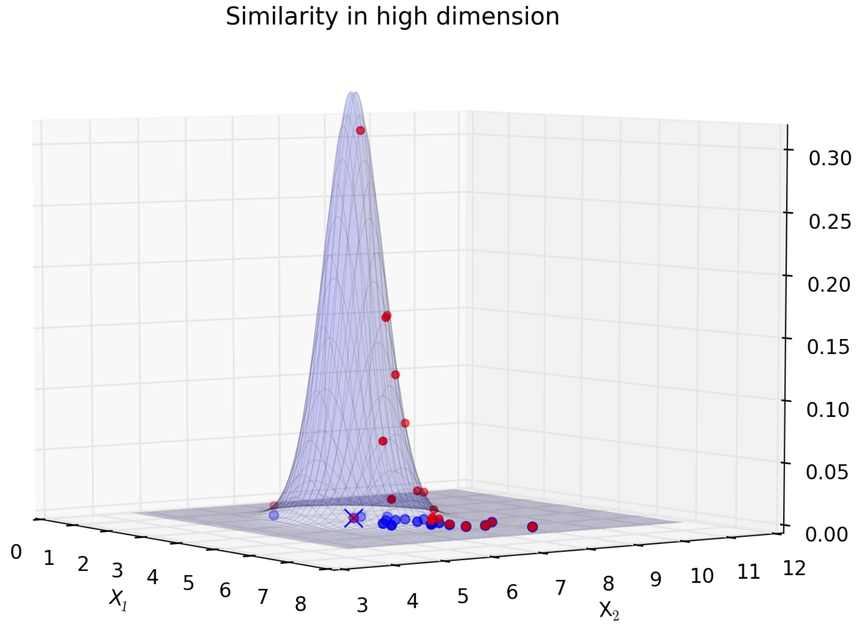
Some questions you might have

- Why is the neighborhood in high D a Gaussian probability density function?
- Why is the neighborhood in low D a Student's t probability density function?
- Why is it called Students t? (SIDEBAR)
- How do you choose σ_i?

- Why is the neighborhood in high D a Gaussian probability density function?

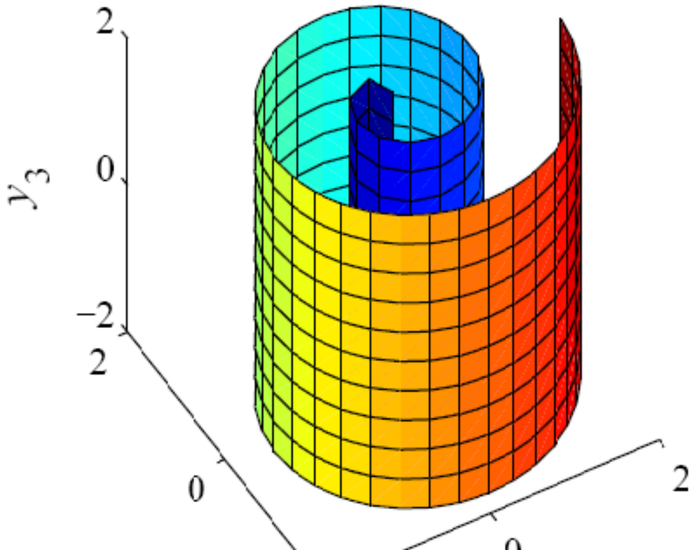
Focus on local geometry.

This is why t-SNE can be interpreted as topology-based

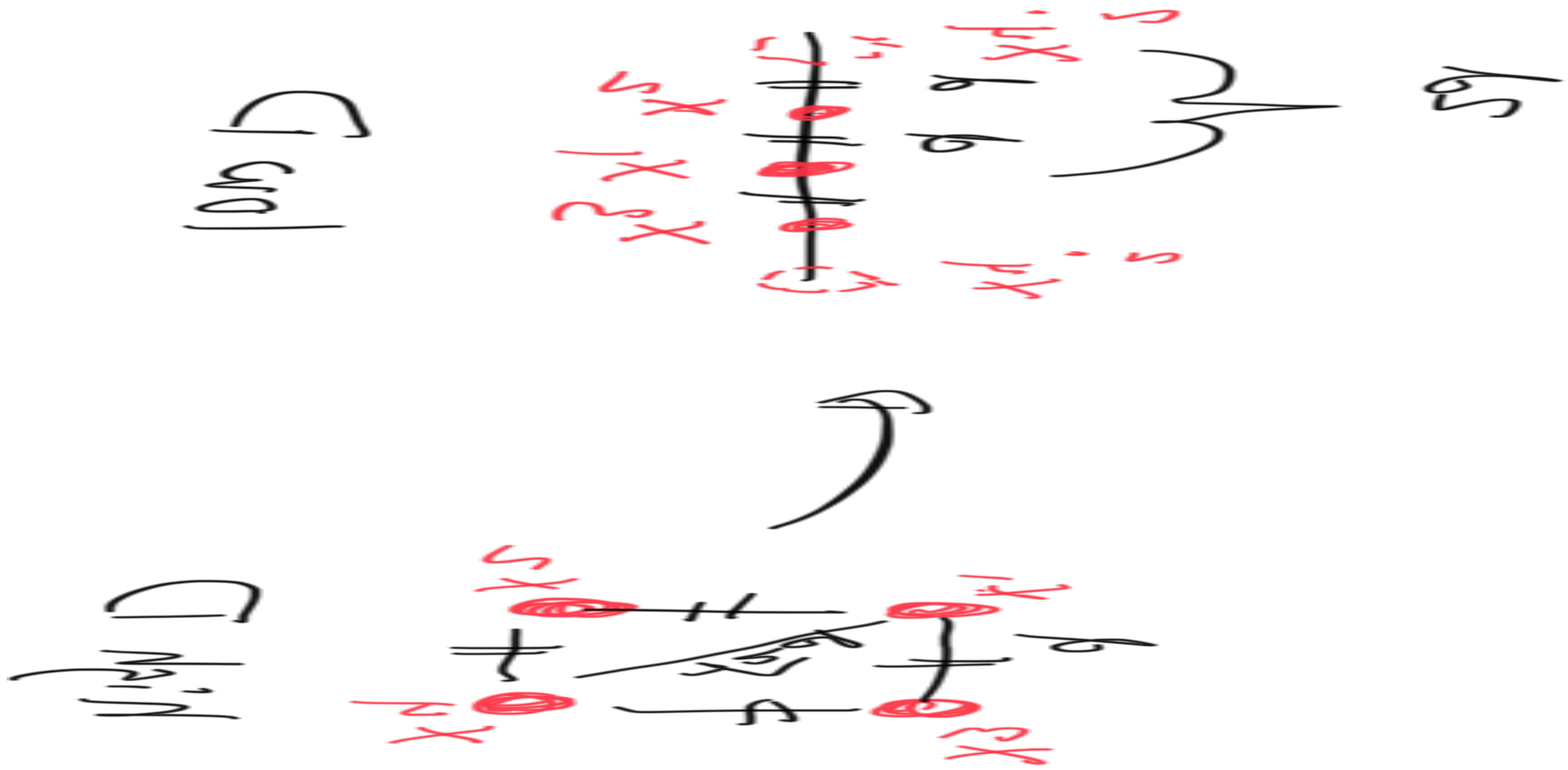


Small distance does not mean proximity on manifold.

Probabilities are appropriate to model this uncertainty

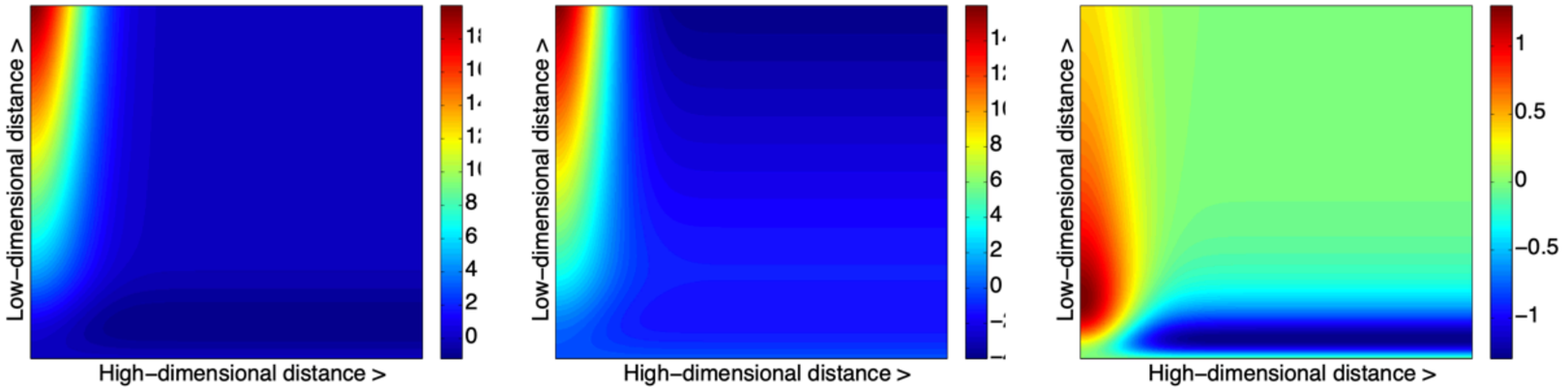


- Why is the neighborhood in low D a Student's t probability density function?



The "crowding problem"…. There's not enough room to accommodate all the neighbors in high D. Long tails help deal with misalignments and conflicts, allowing more wiggle room in placement

Mismatched Tails can Compensate for Mismatched Dimensionalities



(a) Gradient of SNE. (b) Gradient of UNI-SNE. (c) Gradient of t-SNE.

Figure 1: Gradients of three types of SNE as a function of the pairwise Euclidean distance between two points in the high-dimensional and the pairwise distance between the points in the low-dimensional data representation.

- Why is it called Students t? (SIDEBAR)

Because Guinness didn’t want other breweries to know their trade secrets

William Sealy Gosset (13 June 1876 – 16 October 1937) was an English statistician, chemist and brewer who served as Head Brewer of Guinness and Head Experimental Brewer of Guinness and was a pioneer of modern statistics. He pioneered small sample experimental design and analysis with an economic approach to the logic of uncertainty. Gosset published under the pen name **Student** and developed most famously Student's t-distribution – originally called Student's "z" – and "Student's test of statistical significance".[1]

Life and career [edit]

Born in Canterbury, England the eldest son of Agnes Sealy Vidal and Colonel Frederic Gosset, R.E. Royal Engineers, Gosset attended Winchester College before matriculating as Winchester Scholar in natural sciences and mathematics at New College, Oxford. Upon graduating in 1899, he joined the brewery of Arthur Guinness & Son in Dublin, Ireland; he spent the rest of his 38-year career at Guinness.[1][2]

Gosset had three children with Marjory Gosset (née Phillpotts). Harry Gosset (1907–1965) was a consultant paediatrician; Bertha Marian Gosset (1909–2004) was a geographer and nurse; the youngest, Ruth Gosset (1911–1953) married the Oxford mathematician Douglas Roaf and had five children.

In his job as Head Experimental Brewer at Guinness, the self-trained Gosset developed new statistical methods – both in the brewery and on the farm – now central to the design of experiments, to proper use of significance testing on repeated trials, and to analysis of economic significance (an early instance of decision theory interpretation of statistics) and more, such as his small-sample, stratified, and repeated balanced experiments on barley for proving the best yielding varieties.[3] Gosset acquired that knowledge by study, by trial and error, by cooperating with others, and by spending two terms in 1906–1907 in the Biometrics laboratory of Karl Pearson.[4] Gosset and Pearson had a good relationship.[4] Pearson helped Gosset with the mathematics of his papers, including the 1908 papers, but had little appreciation of their importance. The papers addressed the brewer's concern with small samples; biometricians like Pearson, on the other hand, typically had hundreds of observations and saw no urgency in developing small-sample methods.[2]

Gosset's first publication came in 1907, "On the Error of Counting with a Haemacytometer," in which – unbeknownst to Gosset aka "Student" – he rediscovered the Poisson distribution.[3] Another researcher at Guinness had previously published a paper containing trade secrets of the Guinness brewery. The

William Sealy Gosset

William Sealy Gosset (aka *Student*) in 1908 (age 32)

Born	13 June 1876 Canterbury, Kent, England
Died	16 October 1937 (aged 61) Beaconsfield, Buckinghamshire, England
Other names	Student
Alma mater	New College, Oxford, Winchester College
Known for	Student's t-distribution, statistical significance, design of experiments, Monte Carlo method, quality control, Modern synthesis, agricultural economics, econometrics
Children	5, including Isaac Henry Gosset
Scientific career	
Institutions	Guinness Brewery

https://lvdmaaten.github.io/publications/papers/JMLR_2014.pdf

How do you choose σ_i?

One last thing we have not yet mentioned how to set the bandwidths σ_i for the Gaussian kernels centered over each data point in the input space. It is unlikely that one single value of σ_i is optimal for all data points because the density of the data is likely to vary. In dense regions, a smaller value of σ_i is usually more appropriate than in sparser regions. Perplexity is defined as

$$\text{Perplexity}(\mathbf{p}_i) = 2^{H(\mathbf{p}_i)}$$

where H is the Shannon entropy of a discrete distribution

$$H(\mathbf{p}_i) = -\sum_i p_{j|i} \log_2(p_{j|i})$$

Perplexity can be thought of as a continuous analogue to the k nearest neighbours, to which t-SNE will attempt to preserve distances. More concretely, the bandwidths σ_i are set such that each Gaussian kernel fits k nearest neighbors within one standard deviation of the probability density.

- Low Perplexity is small σ_i
- High Perplexity is large σ_i

https://lvdmaaten.github.io/publications/papers/JMLR_2014.pdf

t-SNE algorithm

Due to the asymmetry of the Kullback-Leibler divergence, the objective function focuses on modeling high values of p_{ij} (similar objects) by high values of q_{ij} (nearby points in the embedding space). The objective function is non-convex in the embedding $\mathcal{E}$. It is typically minimized by descending along the gradient:

$$\frac{\partial C}{\partial \mathbf{y}_i} = 4 \sum_{j \neq i} (p_{ij} - q_{ij}) q_{ij} Z (\mathbf{y}_i - \mathbf{y}_j), \tag{5}$$

where we defined the normalization term $Z = \sum_{k \neq l} (1 + \|\mathbf{y}_k - \mathbf{y}_l\|^2)^{-1}$.

It is straightforward to see that the evaluation of the joint distributions P and Q is $\mathcal{O}(N^2)$, because both distributions involve a normalization term that sum over all $N(N-1)$ pairs of unique objects. Since t-SNE scales quadratically in the number of objects N, its applicability is limited to data sets with only a few thousand input objects; beyond that, learning becomes too slow to be practical (and the memory requirements become too large).

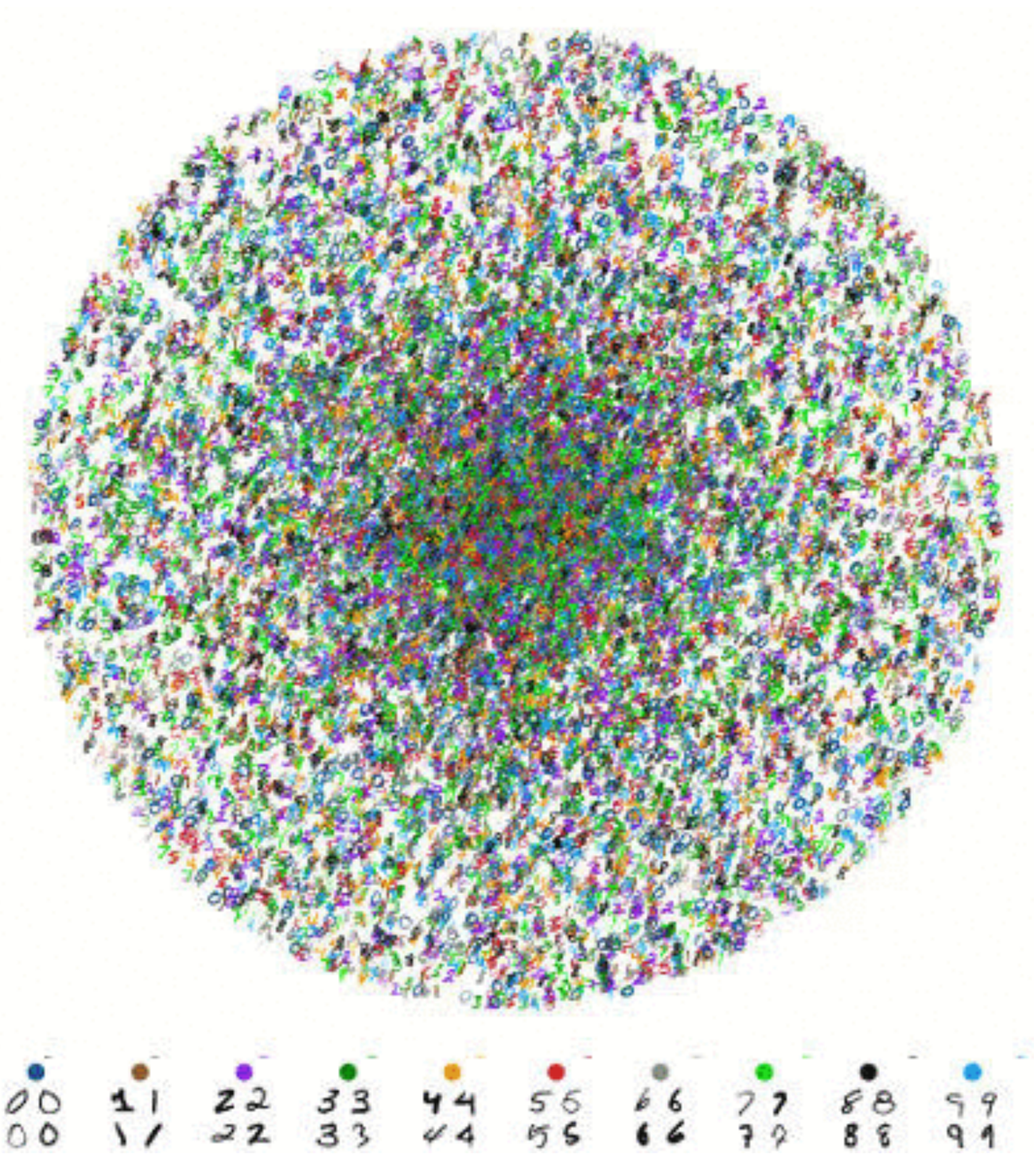
Force directed graph layout

Another way to interpret t-SNE and similar algos

In the neighborhood graph, imagine a set of attractive forces along edges and repulsive forces among vertices

$$\frac{\partial C}{\partial \mathbf{y}_i} = 4 \sum_{j \neq i} (p_{ij} - q_{ij}) q_{ij} Z (\mathbf{y}_i - \mathbf{y}_j),$$

$$\frac{\partial C}{\partial \mathbf{y}_i} = 4(F_{attr} + F_{rep}) = 4 \left(\sum_{j \neq i} p_{ij} q_{ij} Z(\mathbf{y}_i - \mathbf{y}_j) - \sum_{j \neq i} q_{ij}^2 Z(\mathbf{y}_i - \mathbf{y}_j) \right)$$



Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

Data: data set $\mathcal{X} = \{x_1, x_2, ..., x_n\}$,
cost function parameters: perplexity $Perp$,
optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, ..., y_n\}$.
begin
    compute pairwise affinities $p_{j|i}$ with perplexity $Perp$ (using Equation 1)
    set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
    sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, ..., y_n\}$ from $\mathcal{N}(0, 10^{-4}I)$
    for *t=1* **to** T **do**
        compute low-dimensional affinities q_{ij} (using Equation 4)
        compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
        set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t) \left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)} \right)$
    end
end

From SNE to t-SNE.

SNE

Modelisation:

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

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

Cost Function:

$$C = \sum_i KL(P_i||Q_i)$$

Derivatives:

$$\frac{dC}{dy_i} = 2\sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j)$$

$\Rightarrow$

Symmetric SNE

Modelisation:

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

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

Cost Function:

$$C = KL(P||Q)$$

Derivatives:

$$\frac{dC}{dy_i} = 4\sum_j (p_{ij} - q_{ij})(y_i - y_j)$$

- **Faster Computation**

$\Rightarrow$

t-SNE

Modelisation:

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

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

Cost Function:

$$C = KL(P||Q)$$

Derivatives:

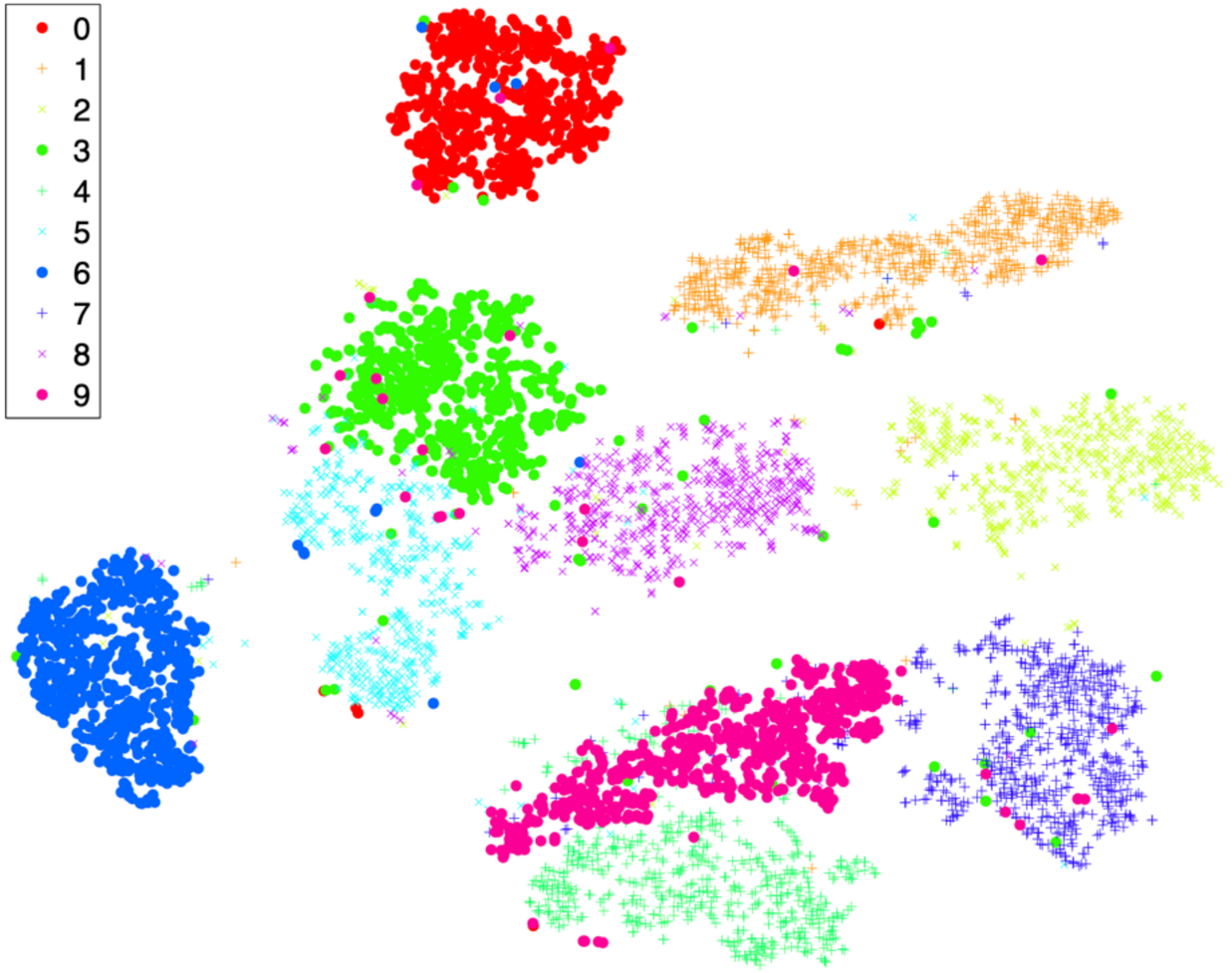
$$\frac{dC}{dy_i} = 4\sum_j (p_{ij} - q_{ij})(y_i - y_j)(1+\| y_i - y_j \|^2)^{-1}$$

- **Even Faster Computation**
- **Better Behaviour**

t-SNE limitations

- Computationally expensive!
  - $\mathcal{O}(N^2)$ because of affinity matrix calculations… and worse than that because we have to do this iteratively on the lower-d embedding
  - Barnes-Hut approximation to gradient descent on KL (a spanning tree style algorithm) compares points to “cells” in $\mathcal{O}(N \log N)$.
    - Can only go up to 3D! Only works with dense input data!
    - Can go up to ~10^5 to 10^6 samples. MNIST able
- Only has .fit_transform() because there’s no way to .transform() separately from a .fit()! You’d have to redo from scratch to calculate where a new data point should go!
  - This technique is for visualization, not for building cross-validated classifiers on top of

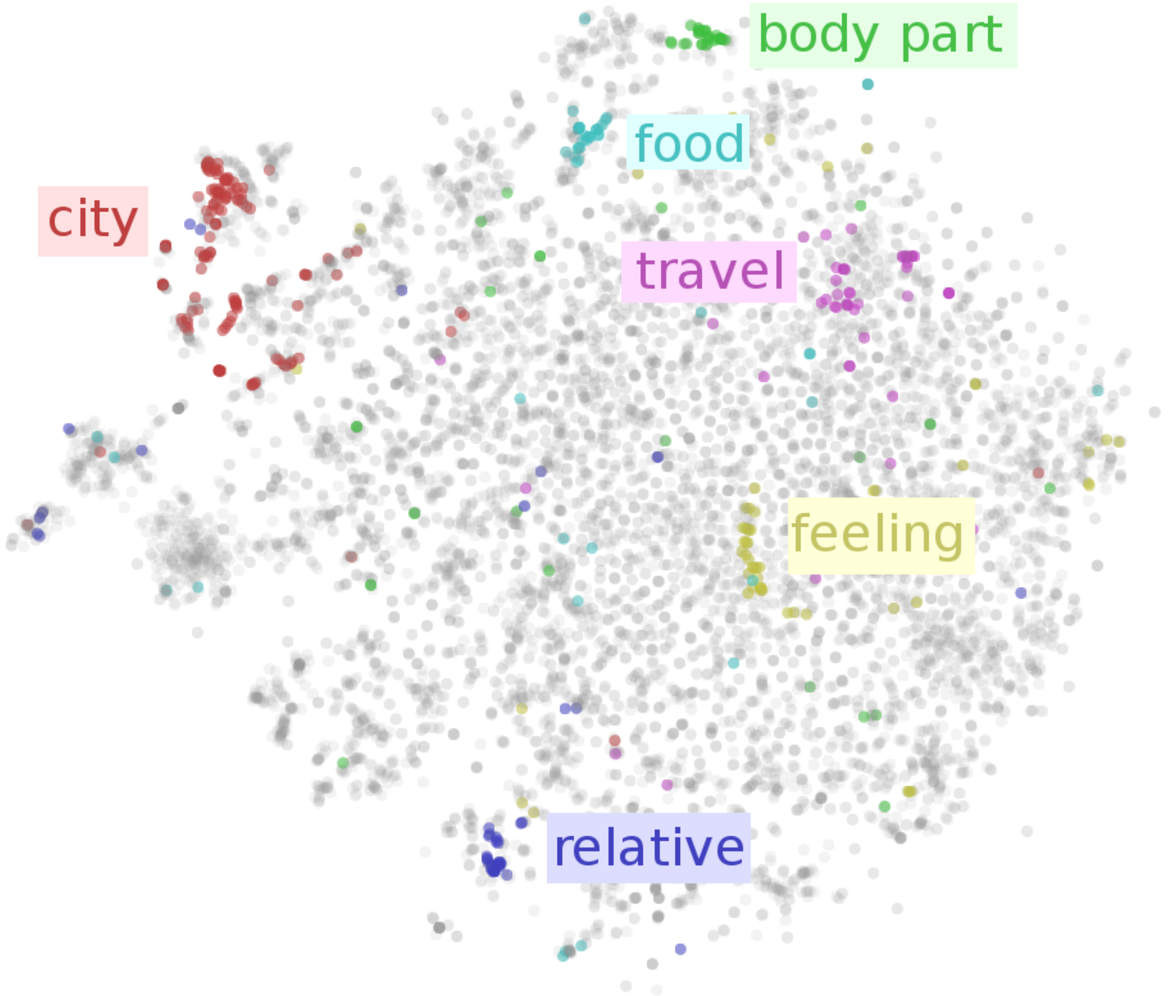
MNIST



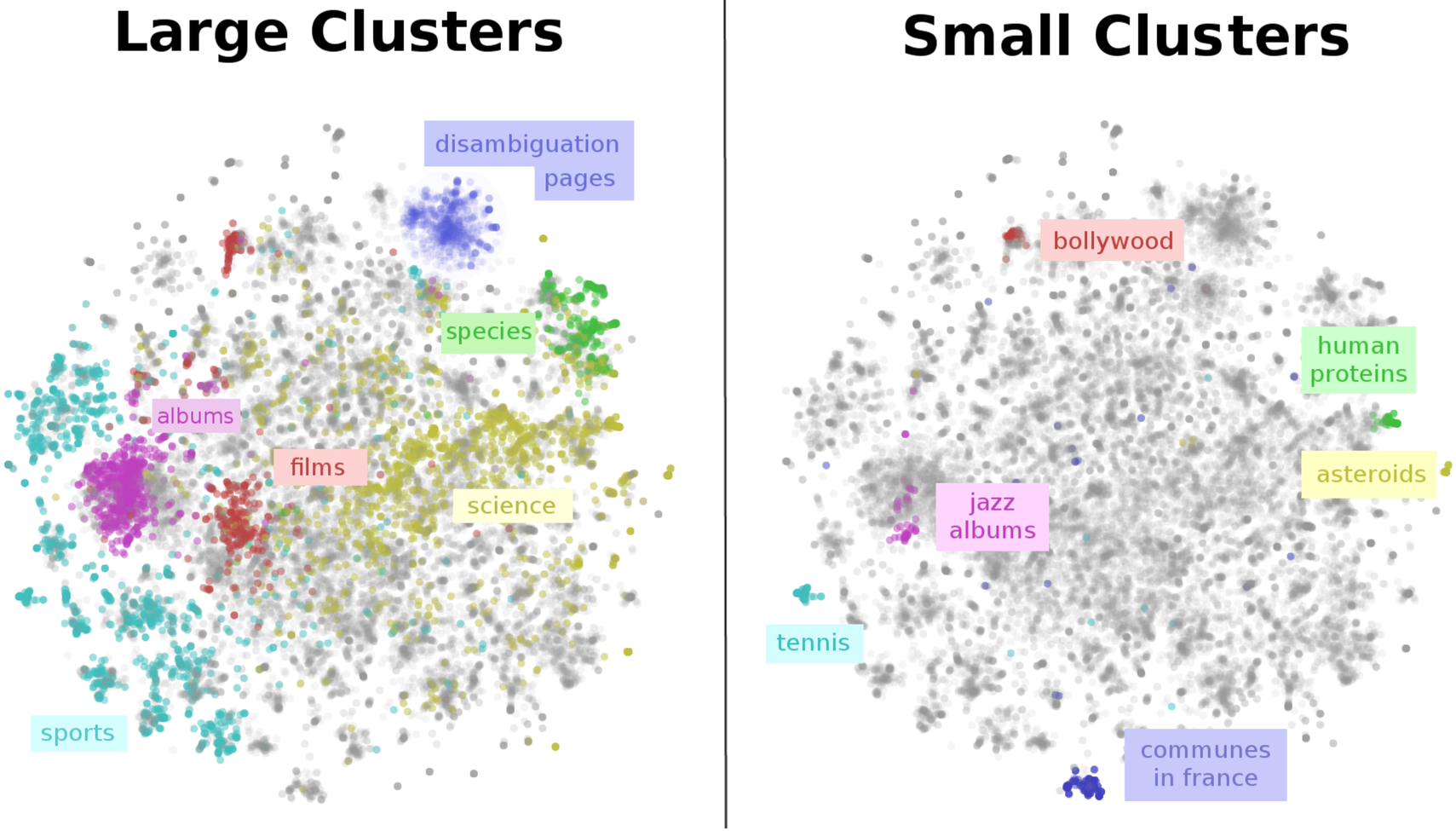
(a) Visualization by t-SNE.

Using t-SNE (or other manifold learning)

Using t-SNE to explore a Word embedding.



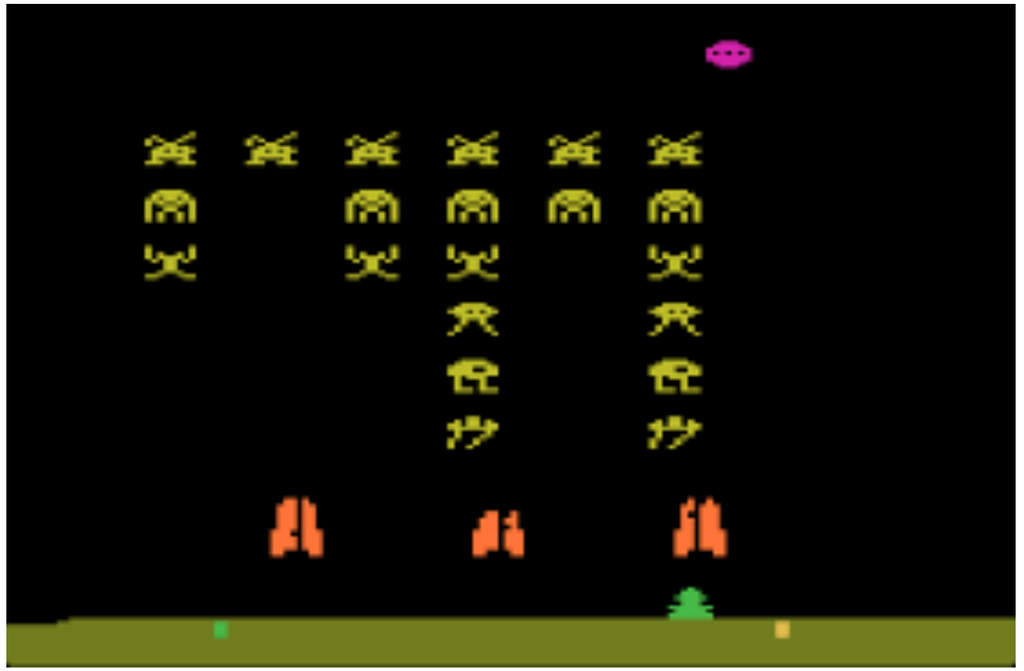
Explore a Wikipedia article embedding.



http://colah.github.io/

Exploring game state representations.

Google Deepmind plays Atari games.



Playing Atari with deep reinforcement learning, V. Mnih et Al.

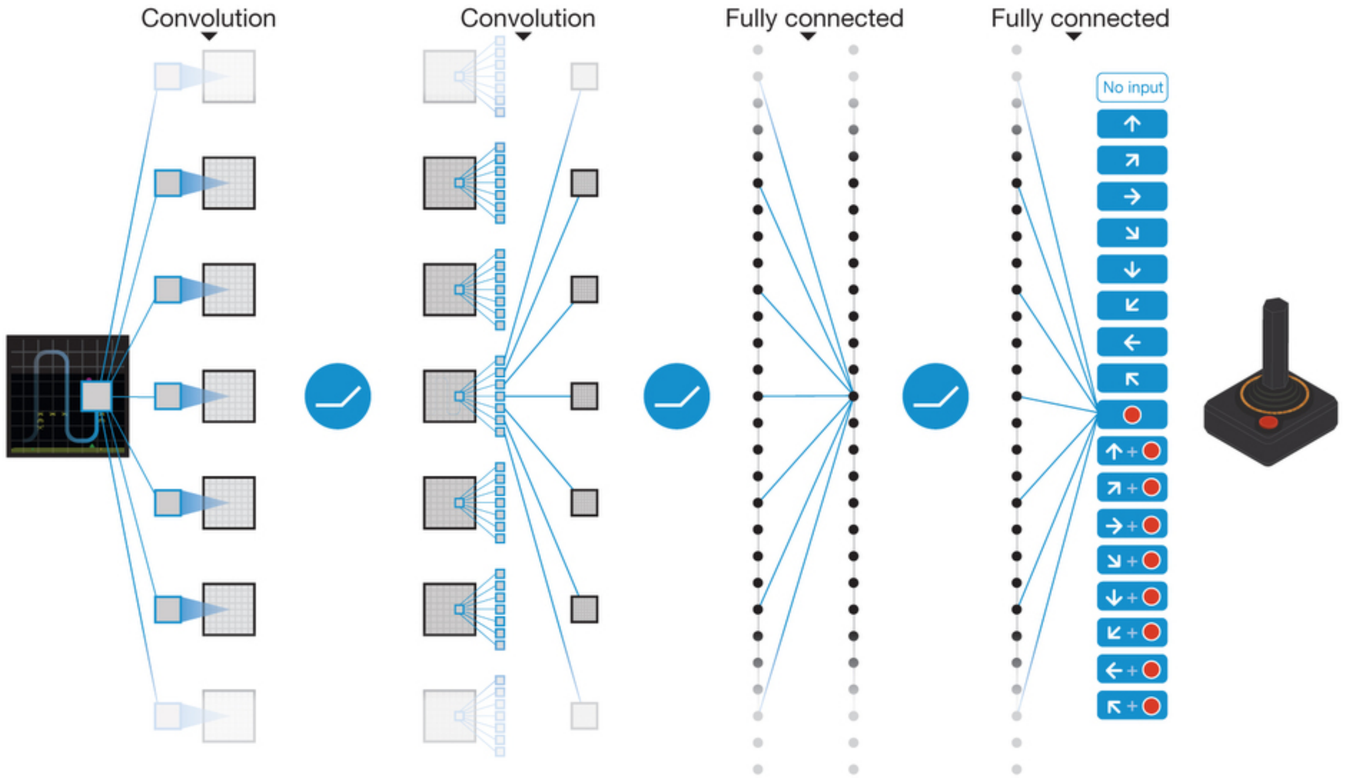
Goal

Learning to play Space Invaders from score feedback and raw pixel values.

Exploring game state representations.

Google Deepmind plays Atari games.

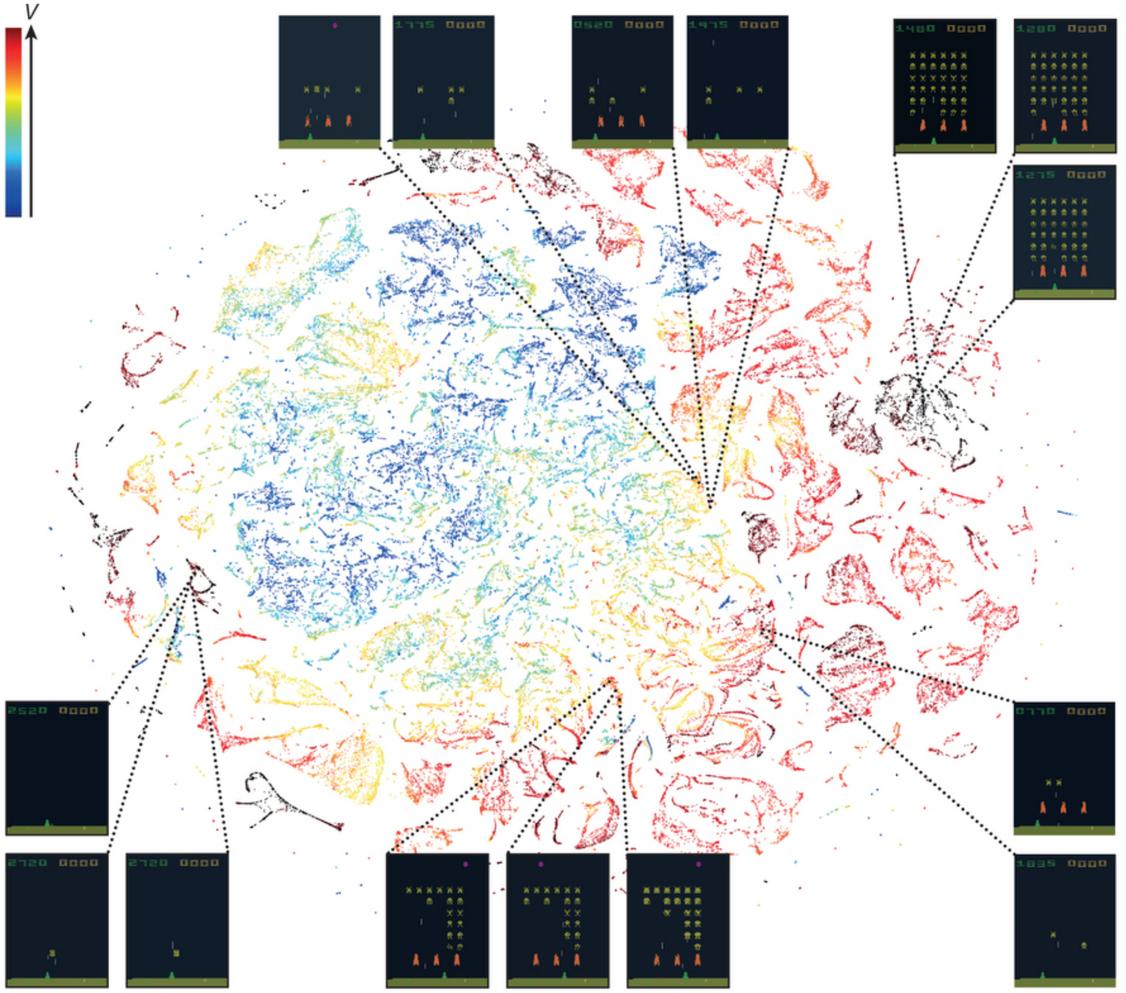
- A representation is learned with a convolutional neural network
- From 84x84x4 = 28.224 pixel values to 512 neurons.
- Predicts expected score if a certain action is taken.



Human-level control through deep reinforcement learning, V. Mnih et Al. (Nature,2015)

Exploring game state representations.

Google Deepmind plays Atari games.

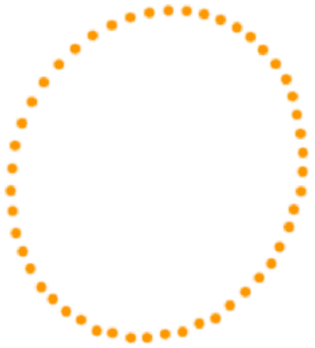


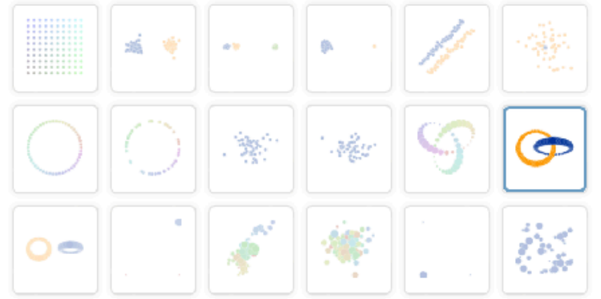
Human-level control through deep reinforcement learning, V. Mnih et Al. (Nature,2015)

https://distill.pub/2016/misread-tsne/

How to Use t-SNE Effectively

Although extremely useful for visualizing high-dimensional data, t-SNE plots can sometimes be mysterious or misleading. By exploring how it behaves in simple cases, we can learn to use it more effectively.





Step
44

Points arranged in 3D, on two linked circles. Different runs may give different results.

Share this view

Number Of Points 50

Perplexity 10

Epsilon 5