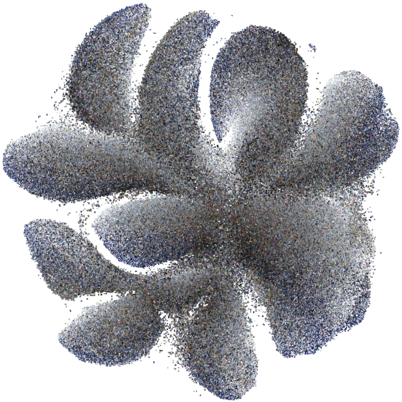
Visualizing Data using t-SNE

Laurens van der Maaten and Geoffrey Hinton, JMLR 2008

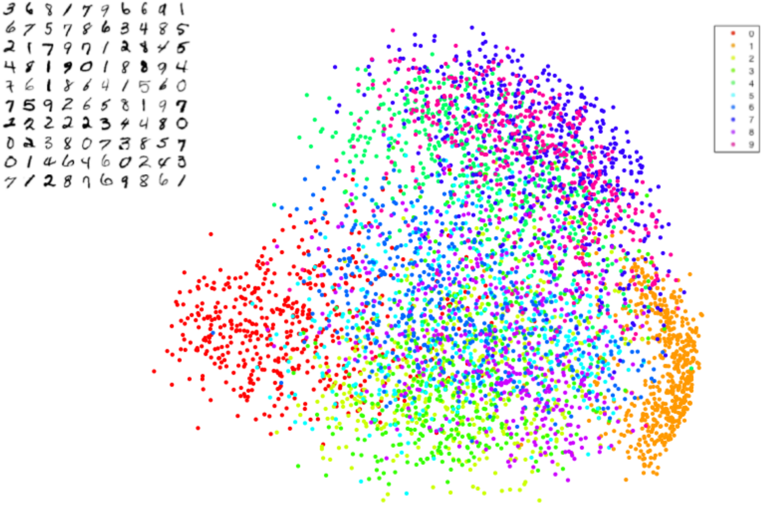
Overview

Overview

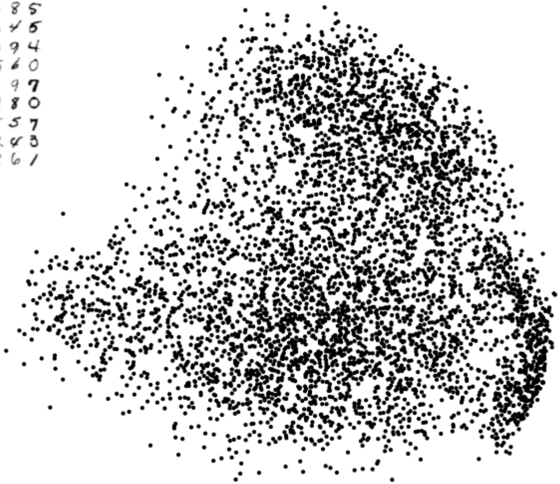
- We are given a collection of N high-dimensional objects $x_1, ... x_N$
- How can we get a feel for how these objects are arranged in the data space?



Principal Components Analysis

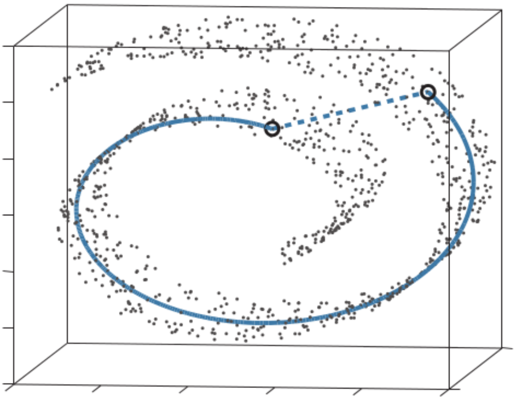


Principal Components Analysis



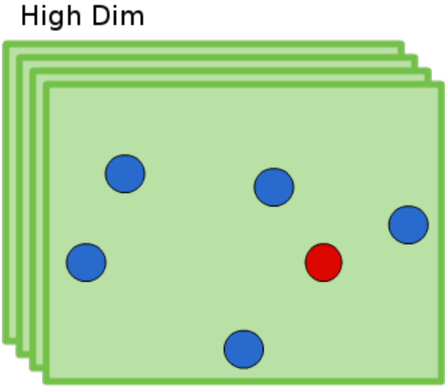
Swiss Roll

- PCA is mainly concerned dimensionality, with preserving when large pairwise distances in the map

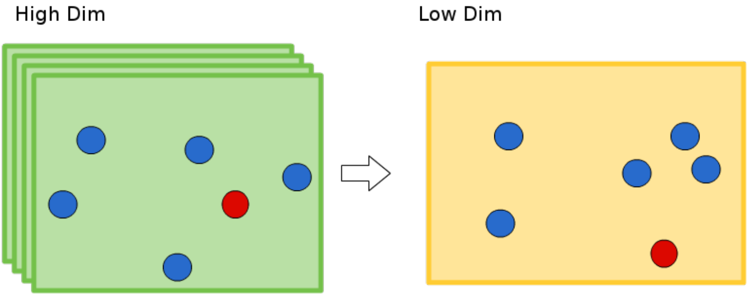


Introduction

- Distance Perservation
- Neighbor Perservation

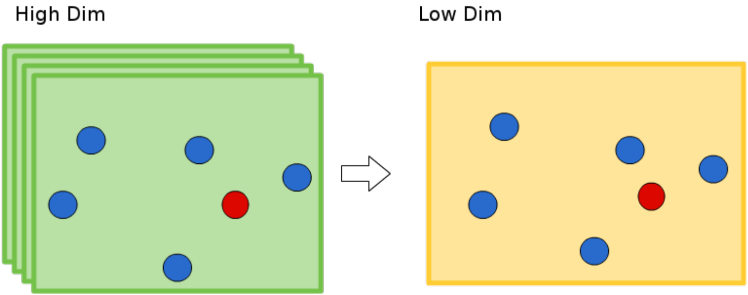


Introduction



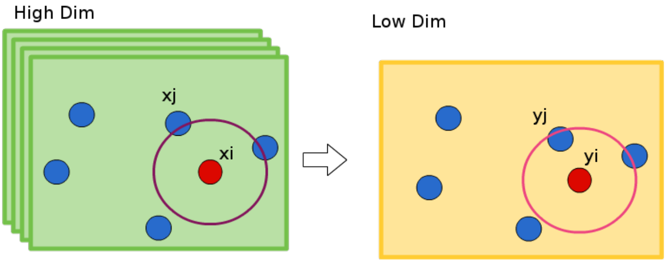
Introduction

Preserve the neighborhood



Introduction

Measure pairwise similarities between high-dimensional and low-dimensonal objects



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

Stochastic Neighbor Embedding

Converting the high-dimensional Euclidean distances into conditional probabilities that represent similarities

- Similarity of datapoints in High Dimension

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

- Similarity of datapoints in Low Dimension

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

- Cost function

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

Minimize the cost function using gradient descent

Stochastic Neighbor Embedding

Gradient has a surprisingly simple form

$$\frac{\partial C}{\partial y_i} = \sum_{j \neq i} (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j)$$

The gradient update with momentum term is given by

$$Y^{(t)} = Y^{(t-1)} + \eta \frac{\partial C}{\partial y_i} + \beta(t)(Y^{(t-1)} - Y^{(t-2)})$$

Symmetric SNE

- Minimize the sum of the KL divergences between the conditional probabilities

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

- Minimize a single KL divergence between a joint probability distribution

$$C = KL(P||Q) = \sum_i \sum_{j \neq i} p_{ij} log \frac{p_{ij}}{q_{ij}}$$

- The obvious way to redefine the pairwise similarities is

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

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

Symmetric SNE

Such that $p_{ij} = p_{ji}, q_{ij} = q_{ji}$, the main advantage is simplifing the gradient

$$\frac{\partial C}{\partial y_i} = 2 \sum_j (p_{ij} - q_{ij})(y_i - y_j)$$

However, in practice we symmetrize (or average) the conditionals

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

Set the bandwidth σ_i such that the conditional has a fixed perplexity (effective number of neighbors) $Perp(P_i) = 2^{H(P_i)}$, typical value is about 5 to 50

t-Distribution

Use heavier tail distribution than Gaussian in low-dim space, we choose

$$q_{ij} \propto (1 + ||y_i - y_j||^2)^{-1}$$

Then the gradient could be

$$\frac{\partial C}{\partial y_i} = 4 \sum_{j \neq i} (p_{ij} - q_{ij})(1 + ||y_i - y_j||^2)^{-1}(y_i - y_j)$$

t-Distributed Stochastic Neighbor Embedding

- Similarity of datapoints in High Dimension

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

- Similarity of datapoints in Low Dimension

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

t-Distributed Stochastic Neighbor Embedding

- Cost function

$$C = KL(P||Q) = \sum_i \sum_j p_{ij} log \frac{p_{ij}}{q_{ij}}$$

  - Large p_{ij} modeled by small q_{ij}: Large penalty
  - Small p_{ij} modeled by large q_{ij}: Small penalty
  - t-SNE mainly preserves local similarity structure of the data

- Gradient

$$\frac{\partial C}{\partial y_i} = 4 \sum_{j \neq i} (p_{ij} - q_{ij})(1 + ||y_i - y_j||^2)^{-1}(y_i - y_j)$$

Gradient Interpretation

Pairwise Euclidean distance between two points in the high-dim and in low-dim data representation

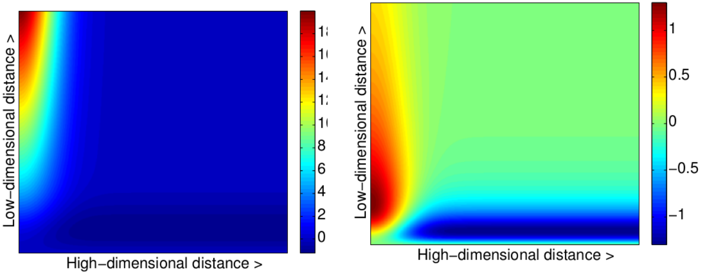
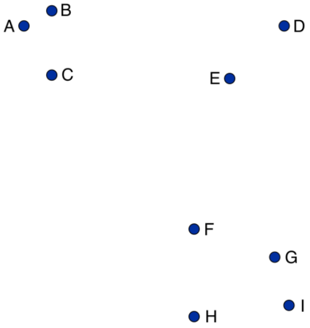


Figure : Gradient of SNE and t-SNE

Gradient Interpretation

We can interpret the t-SNE gradient as a simulation of an N-body system

$$\frac{\partial C}{\partial y_i} = 4 \sum_{j \neq i} (p_{ij} - q_{ij})(1 + ||y_i - y_j||^2)^{-1}(y_i - y_j)$$

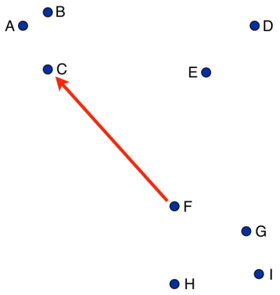


Gradient Interpretation

We can interpret the t-SNE gradient as a simulation of an N-body system

- Displacement

$$(y_i - y_j)$$

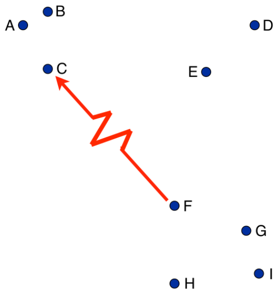


Gradient Interpretation

We can interpret the t-SNE gradient as a simulation of an N-body system

- Exertion / Compression

$$(p_{ij} - q_{ij})(1 + ||y_i - y_j||^2)^{-1}$$

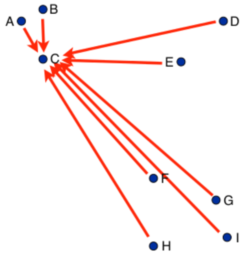


Gradient Interpretation

We can interpret the t-SNE gradient as a simulation of an N-body system

- N-Body, summation

$$\frac{\partial C}{\partial y_i} = 4 \sum_{j \neq i} (p_{ij} - q_{ij})(1 + ||y_i - y_j||^2)^{-1}(y_i - y_j)$$



Reduce Complexity from $O(N^2)$ to $O(N \log N)$ via Barnes Hut (tree-based) algorithm

Experiment & Results

MNIST

- Randomly selected 6,000 images
- $28 \times 28 = 784$ pixels

Olivetti faces

- 400 images (10 per individual)
- $92 \times 112 = 10,304$ pixels

COIL-20

- 20 different objects and 72 equally spaced orientations, yielding a total of 1,440 images
- $32 \times 32 = 1024$ pixels

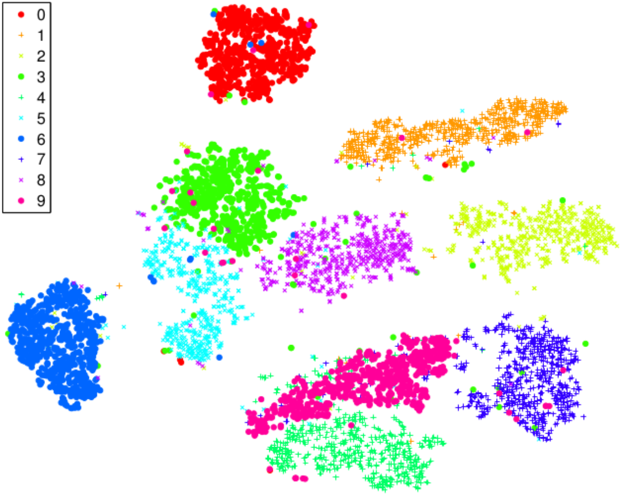
Start by using PCA to reduce the dimensionality of the data to 30

Experiment & Results

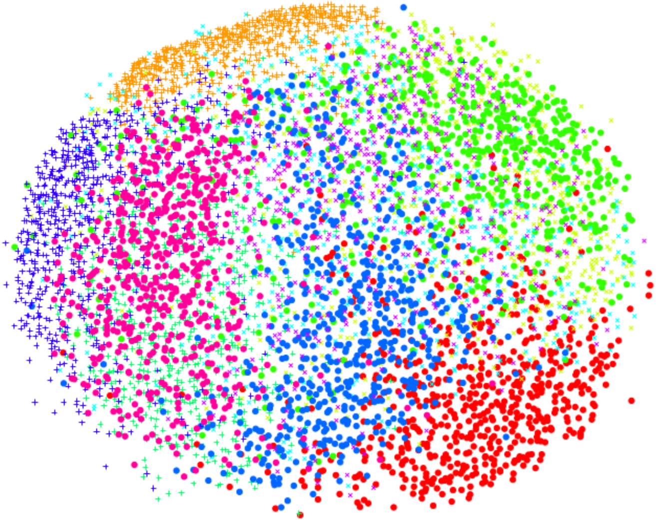
Technique	*Cost function parameters*
t-SNE	$Perp = 40$
Sammon mapping	none
Isomap	$k = 12$
LLE	$k = 12$

Table 1: Cost function parameter settings for the experiments.

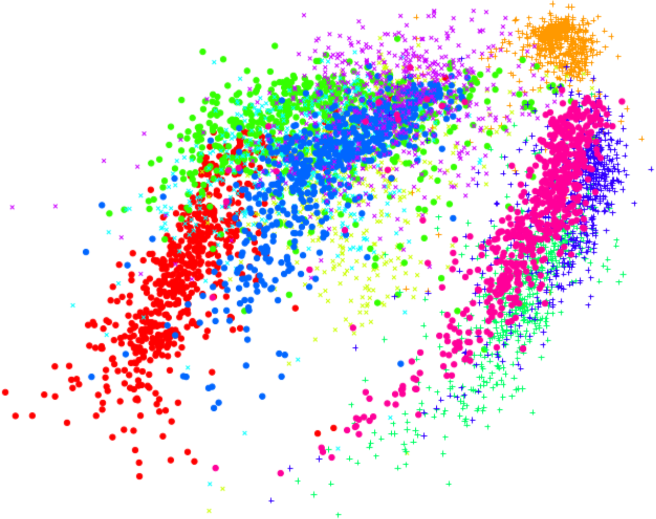
MNIST t-SNE



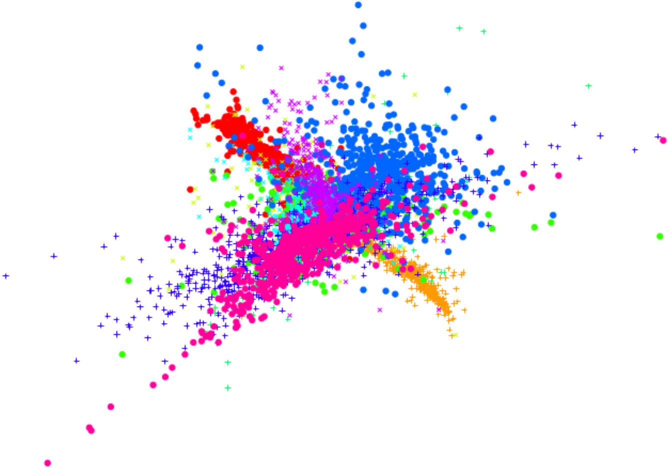
MNIST Sammon



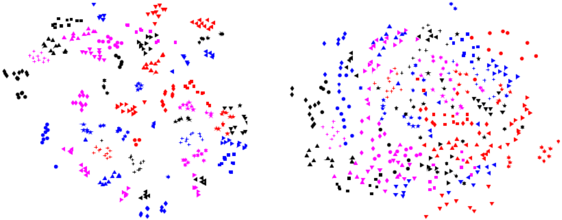
MNIST Isomap



MNIST LLE

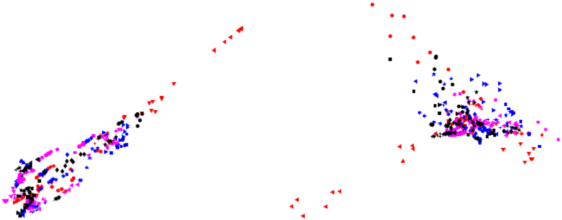


Olivetti faces



(a) Visualization by t-SNE.

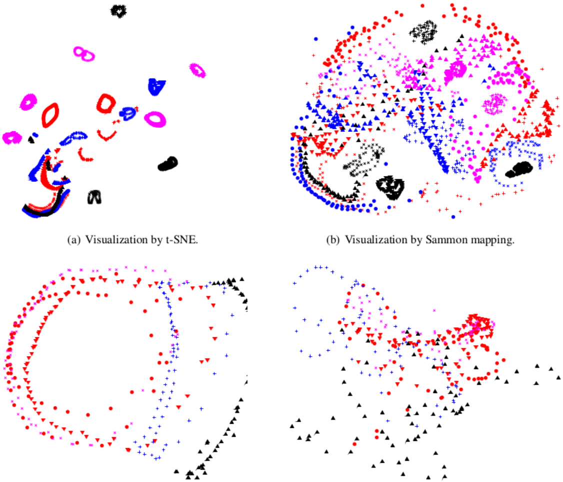
(b) Visualization by Sammon mapping.



(c) Visualization by Isomap.

(d) Visualization by LLE.

COIL-20



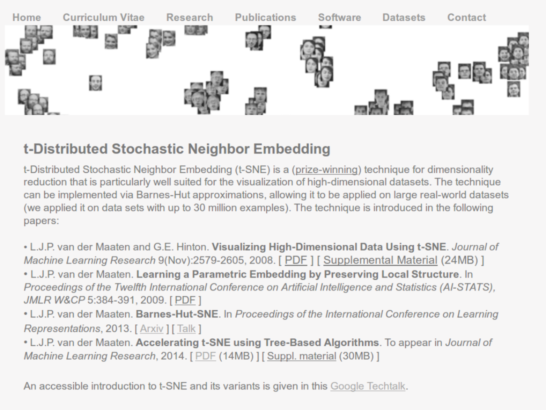
(a) Visualization by t-SNE.

(b) Visualization by Sammon mapping.

(c) Visualization by Isomap.

(d) Visualization by LLE.

Web Resources



Home Curriculum Vitae Research Publications Software Datasets Contact

t-Distributed Stochastic Neighbor Embedding

t-Distributed Stochastic Neighbor Embedding (t-SNE) is a (prize-winning) technique for dimensionality reduction that is particularly well suited for the visualization of high-dimensional datasets. The technique can be implemented via Barnes-Hut approximations, allowing it to be applied on large real-world datasets (we applied it on data sets with up to 30 million examples). The technique is introduced in the following papers:

- L.J.P. van der Maaten and G.E. Hinton. **Visualizing High-Dimensional Data Using t-SNE**. *Journal of Machine Learning Research* 9(Nov):2579-2605, 2008. [PDF] [Supplemental Material (24MB)]
- L.J.P. van der Maaten. **Learning a Parametric Embedding by Preserving Local Structure**. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics (AI-STATS), JMLR W&CP* 5:384-391, 2009. [PDF]
- L.J.P. van der Maaten. **Barnes-Hut-SNE**. In *Proceedings of the International Conference on Learning Representations*, 2013. [Arxiv] [Talk]
- L.J.P. van der Maaten. **Accelerating t-SNE using Tree-Based Algorithms**. To appear in *Journal of Machine Learning Research*, 2014. [PDF (14MB)] [Suppl. material (30MB)]

An accessible introduction to t-SNE and its variants is given in this Google Techtalk.

Google: t-sne

Link: http://homepage.tudelft.nl/19j49/t-SNE.html

Source Codes

- t-SNE (Matlab, CUDA, Binary, Python, Torch, Julia, R and JavaScript)
- Parametric t-SNE (Matlab)
- Barnes-Hut-SNE (with C++, Matlab, Python, Torch, and R wrappers)

Thanks for your patience