Computational Systems Biology
Deep Learning in the Life Sciences

6.802 6.874 20.390 20.490 HST.506

David Gifford
Lecture 5
March 2, 2017

The Zen of PCA, t-SNE, and Autoencoders

Massachusetts Institute of Technology

http://mit6874.github.io

Overall goal for today

- Understand the difference between linear and non-linear manifold embeddings
- Learn the key ideas of Principle Component Analysis, t-SNE, and auto encoders

Today's lecture

- Principle Component Analysis
  - Why do we want to embed data in a lower dimensional space?
  - Discovering a linear embedding that minimizes loss of information
- T-distributed Stochastic Network Embedding (t-SNE)
  - Kullback–Leibler divergence (KL divergence)
  - Minimize D_KL(High || Low)
- Autoencoders
  - Deep learning based encoding in latent space
  - Parameters are optimized to make input and output identical

A **manifold** is a topological space that locally resembles Euclidean space near each point

A **manifold embedding** is a structure preserving mapping of a high dimensional space into a manifold

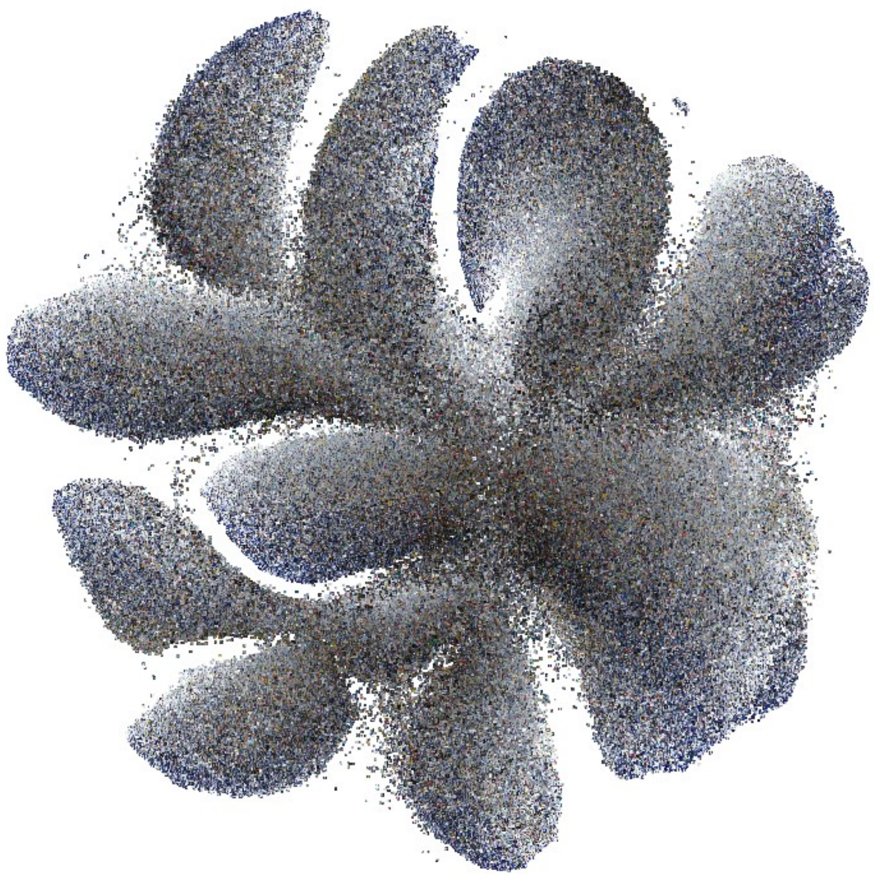
Manifold learning learns a lower dimensional space that enables a manifold embedding

Principle Component Analysis (Part I)

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?



Principle Component Analysis (PCA)

- How can we discover vector components that describe our data?
  1. To discover hidden factors that explain the data
  2. Similar to cluster centroids
  3. To reduce the dimensionality of our data

Principle Component Analysis (PCA)

- Consider the variance of X projected onto vector v

$$\begin{aligned}
Var(v^T X) &= E[(v^T X)^2] - E[v^T X]^2 && (14)\\
&= v^T E[XX^T]v - v^T E[X]E[X^T]v && (15)\\
&= v^T (E[XX^T] - E[X]E[X^T])v && (16)\\
&= v^T \Sigma v && (17)
\end{aligned}$$

- We would like to pick v_i to maximize the variance with the constraint $v_i^T v_i = 1$. Each v_i will be orthogonal to all of the other v_i
- The v_i are called the eigenvectors of Σ and λ_i^2 are the eigenvalues:

$$\begin{aligned}
\Sigma v_i &= \lambda_i^2 v_i && (18)\\
v_i^T \Sigma v_i &= v_i^T \lambda_i^2 v_i && (19)\\
v_i^T \Sigma v_i &= \lambda_i^2 v_i^T v_i && (20)\\
v_i^T \Sigma v_i &= \lambda_i^2 && (21)
\end{aligned}$$

Principle Component Analysis (PCA)

- How do we find the eigenvectors v_i?
- We use singular value decomposition to decompose Σ into an orthogonal rotation matrix U and a diagonal scaling matrix S:

$$\begin{aligned} \Sigma &= USU^T && (22) \\ \Sigma U &= (USU^T)U && (23) \\ &= US && (24) \end{aligned}$$

- The columns of U are the v_i, and S is the diagonal matrix of eigenvalues λ_i^2

Principle Component Analysis (PCA)

- How do we interpret eigenvectors and eigenvalues with respect to our orginal transform A?

$$X = AZ + \mu \tag{25}$$

- A is:

$$A = US^{1/2} \tag{26}$$
$$\Sigma = AA^T \tag{27}$$
$$\Sigma = USU^T \tag{28}$$

- Thus, the transformation A scales by $S^{1/2}$ and rotates by U independent Gaussians to make X

$$Z_i \sim N(0,1) \tag{29}$$
$$X = US^{1/2}Z + \mu \tag{30}$$

Multi-Variate Gaussian Review

- Recall multi-variate Gaussians:

$$
\begin{aligned}
Z_i &\sim N(0,1) &(5)\\
X &= AZ + \mu &(6)\\
\Sigma &= E[(X-\mu)(X-\mu)^T] &(7)\\
&= E[(AZ)(AZ)^T] &(8)\\
&= E[AZZ^T A^T] &(9)\\
&= AE[ZZ^T]A^T &(10)\\
&= AA^T &(11)
\end{aligned}
$$

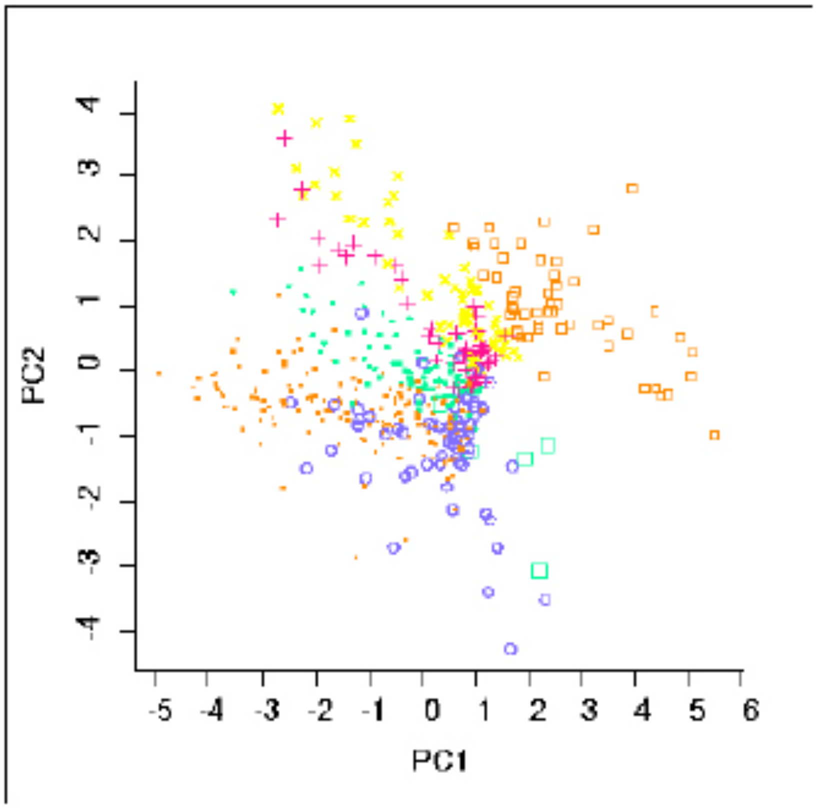
- A multivariate Gaussian model

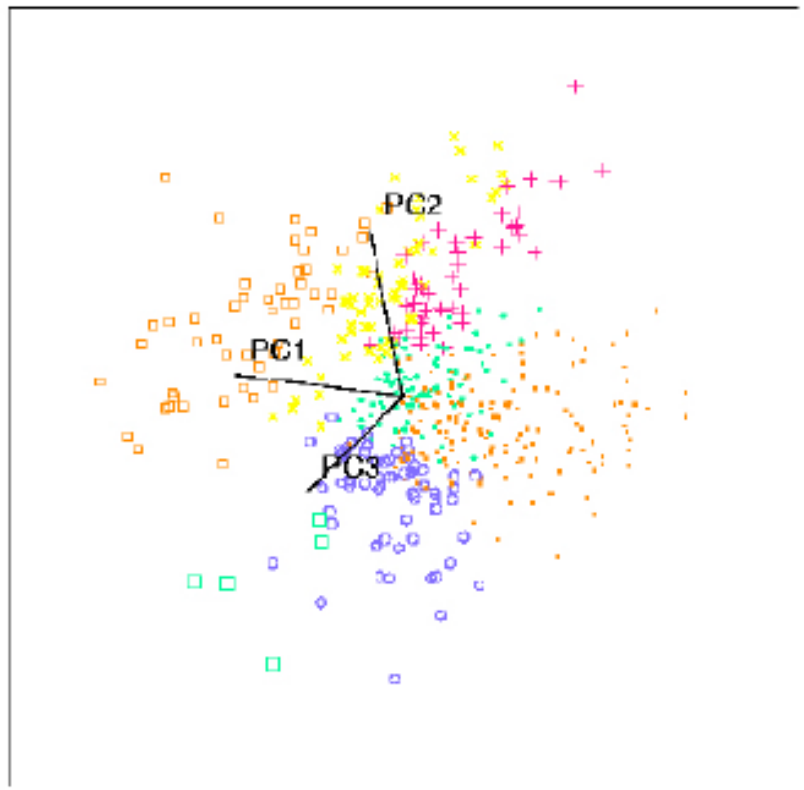
$$
\begin{aligned}
p(x|\theta) &= \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\{\, -\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\,\} &(12)\\
X &\sim N(\mu,\Sigma) &(13)
\end{aligned}
$$

where μ is the mean vector and Σ is the covariance matrix

Example PCA Analysis

477 sporulation genes classified into seven patterns resovled by PCA

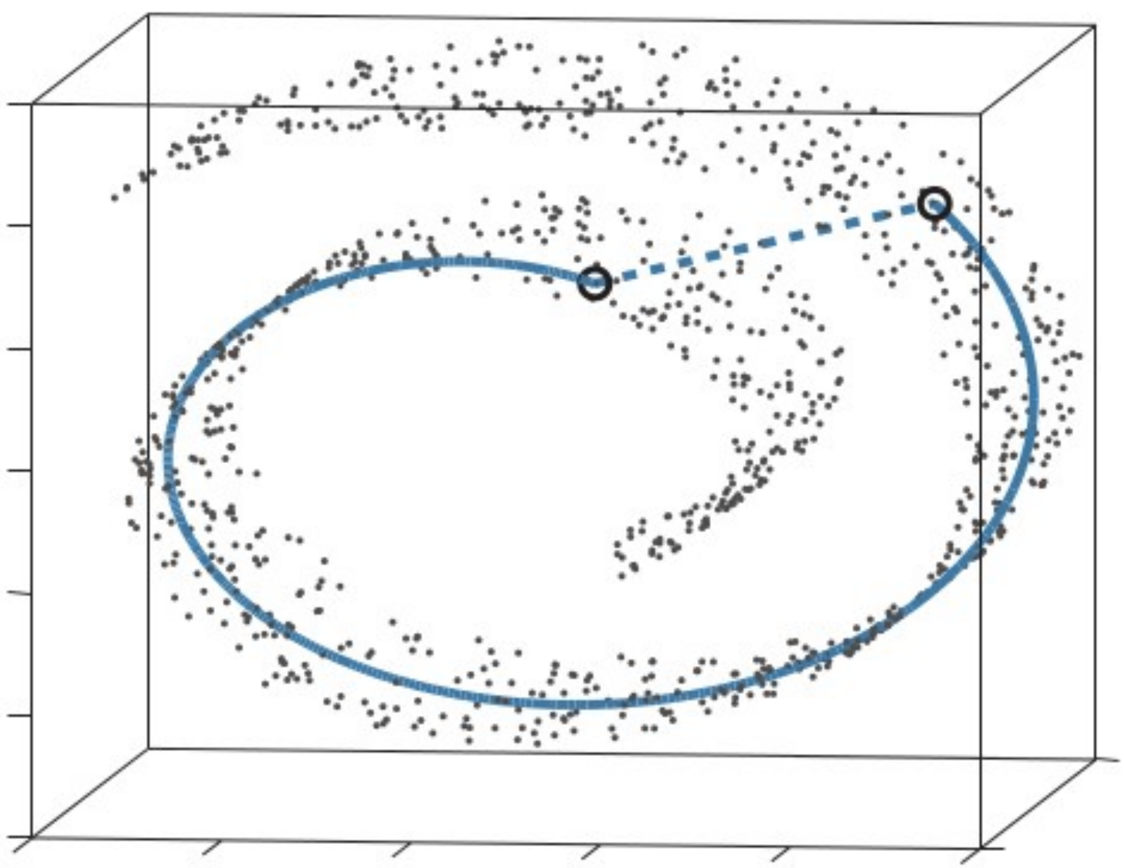




Principal Components Analysis

Swiss Roll

PCA prefers to preserve large pairwise distances in the map as squared distances (variances) overwhelm small distances



t-SNE Multidimensional Scaling
(Part II)

Kullback–Leibler divergence is number of extra bits per sample to encode P using code optimized for Q

$$D_{KL}(P||Q) = \mathbf{E}_{x\sim P}\left[log\frac{P(x)}{Q(x)}\right] \tag{1}$$

$$D_{KL}(P||Q) = \sum_x P(x)\log\frac{P(x)}{Q(x)} \tag{2}$$

$$D_{KL}(P||Q) = -\sum_x P(x)\log Q(x) + \sum_x P(x)\log P(x) \tag{3}$$

$$D_{KL}(P||Q) = H(P,Q) - H(P) \tag{4}$$

KL divergence is asymmetric - using one Gaussian to approximate two Gaussians

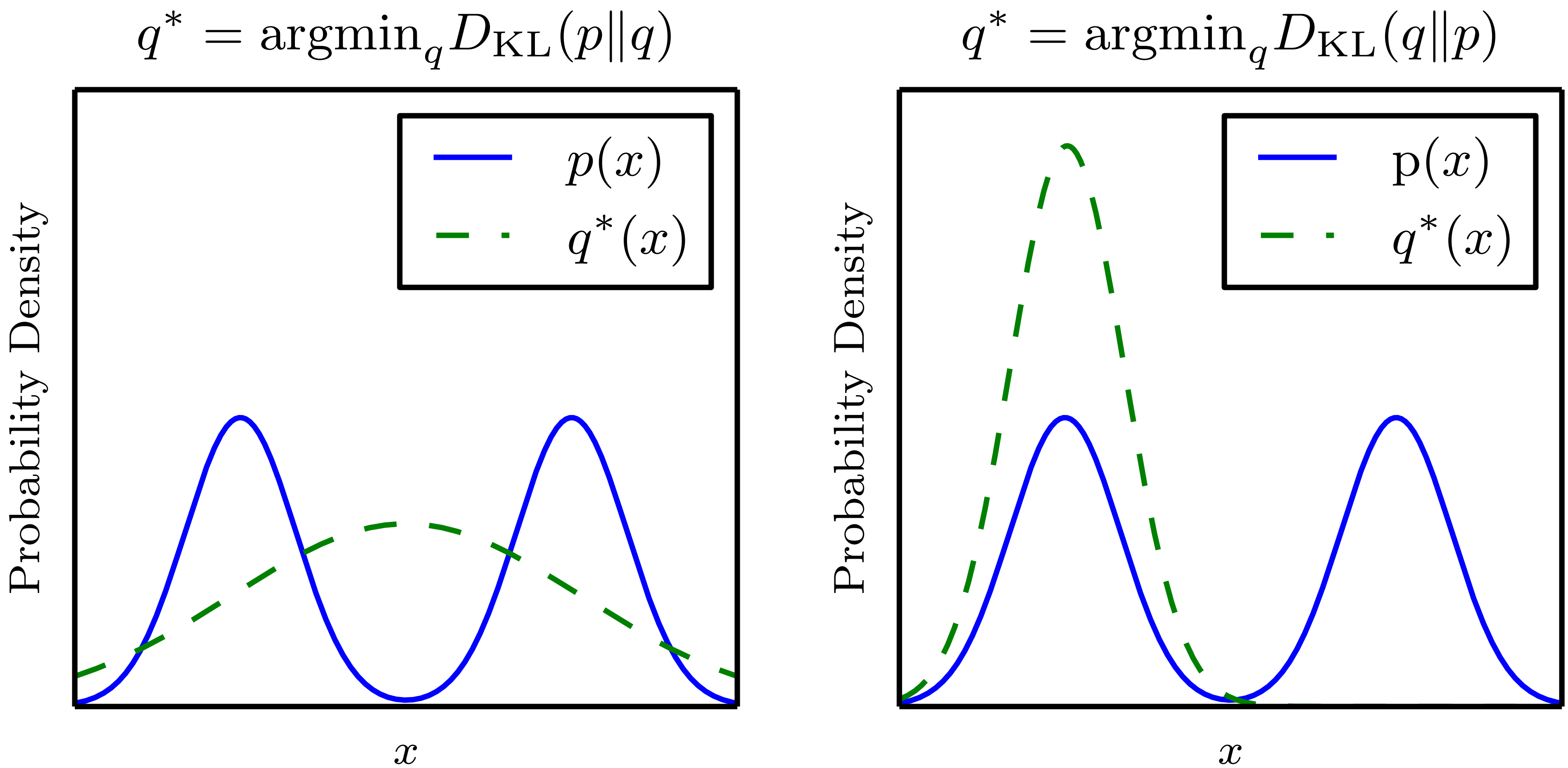
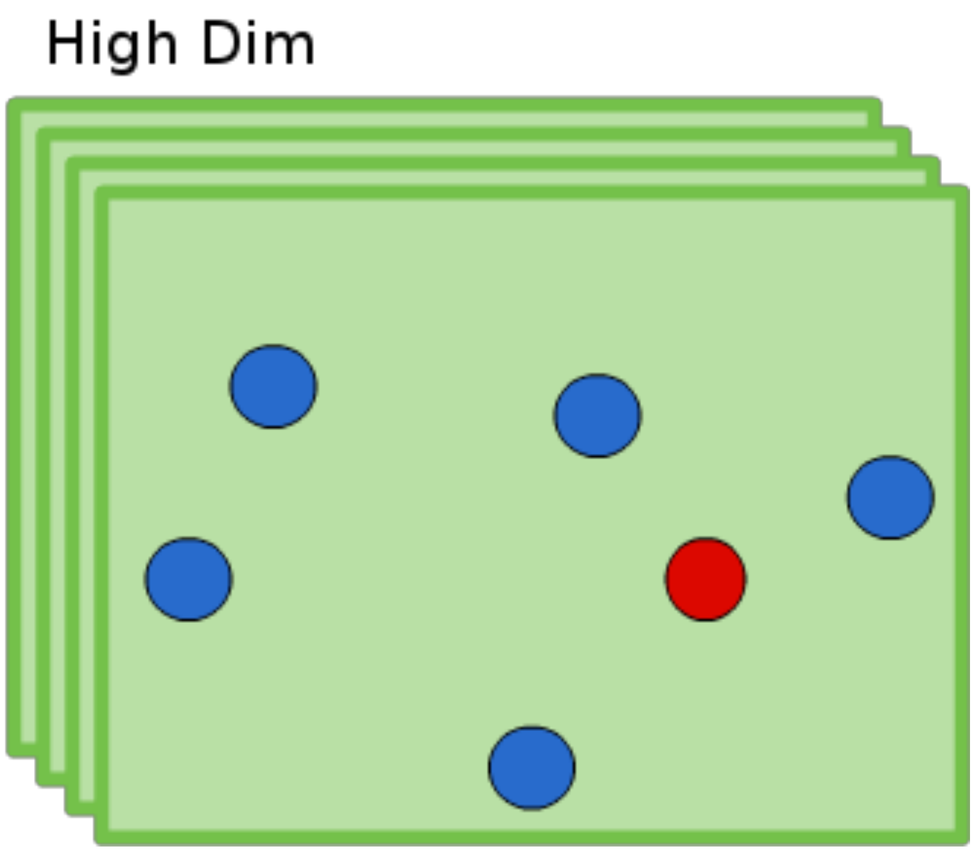


Figure 3.6

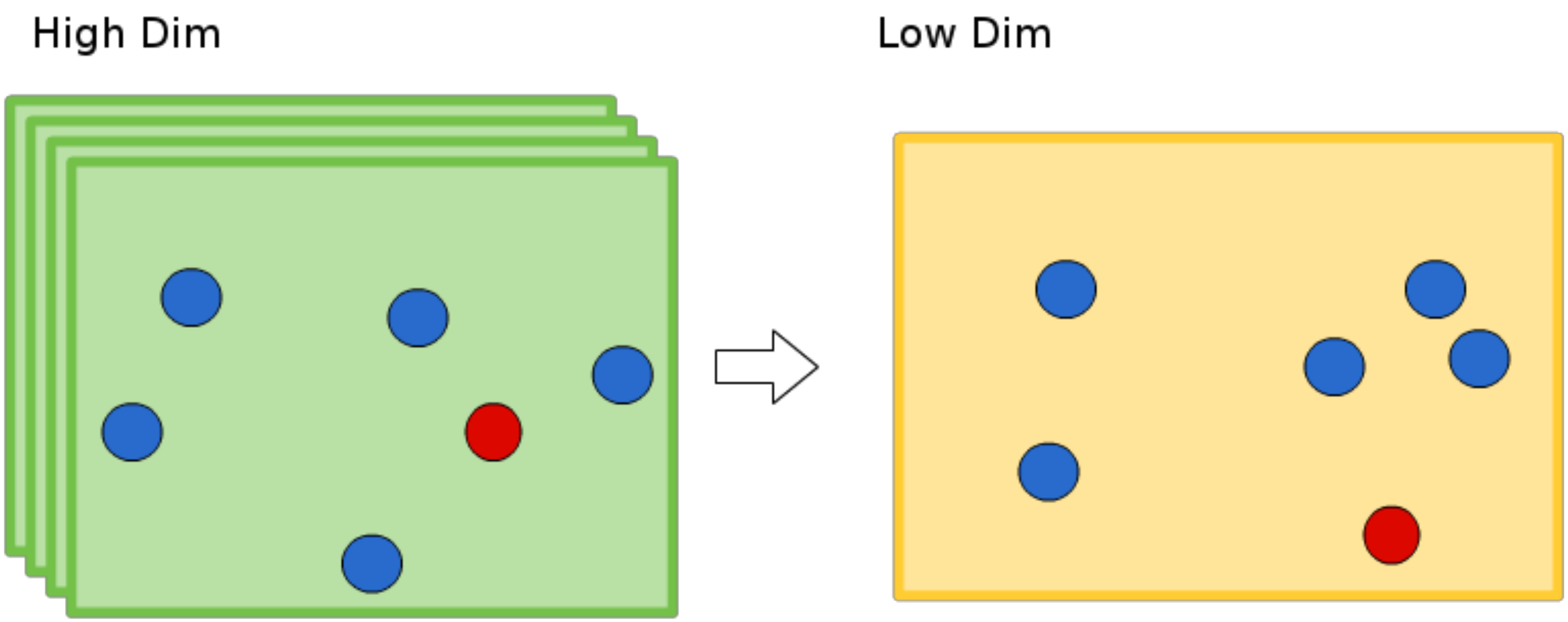
(Goodfellow 2016)

Introduction

- Distance Perservation
- Neighbor Perservation

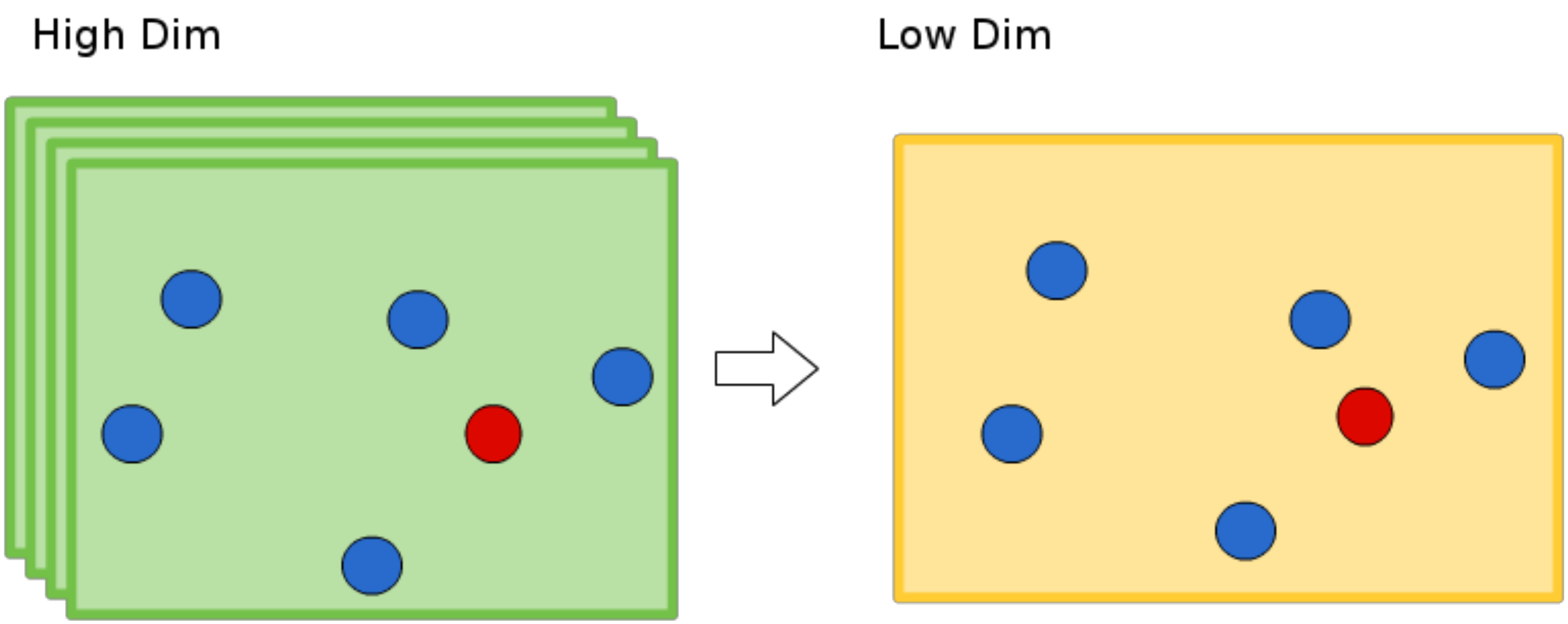


Introduction



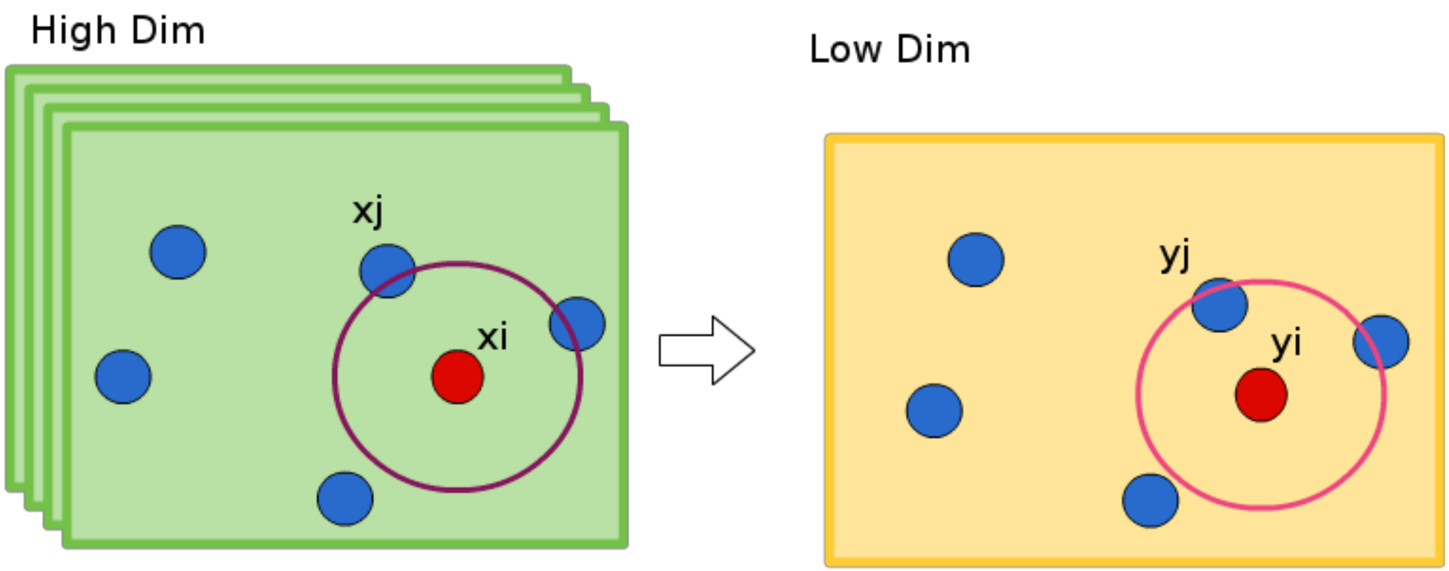
Introduction

Preserve the neighborhood



Introduction

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



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

We will minimize cost based on the divergence of neighborhood probabilities in the higher dimensional space p_{ij} and lower dimensional space q_{ij}

$$C = \sum_i D_{KL}(P_i \| Q_i) \tag{6}$$

$$C = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{7}$$

t-Distributed Stochastic Neighbor Embedding uses the Student t-distribution to avoid overcrowding

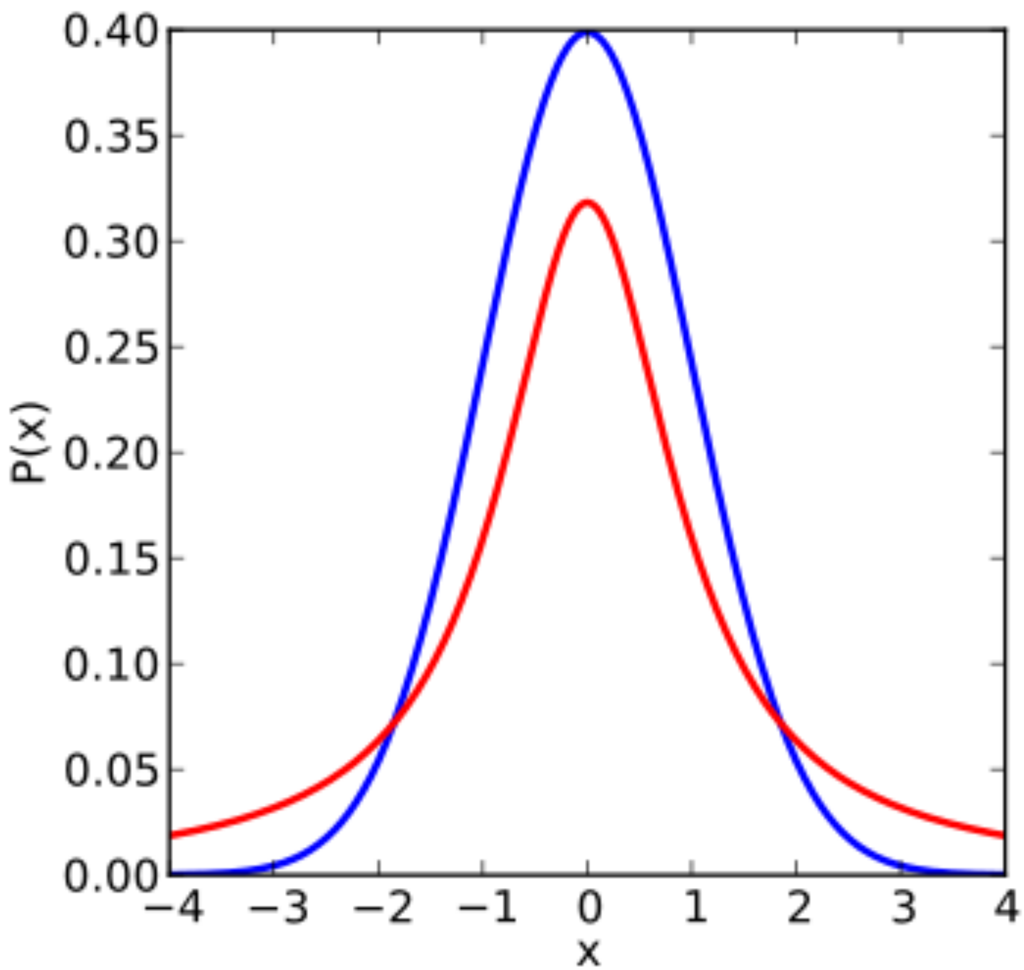
- Similarity of datapoints in High Dimension

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

- Similarity of datapoints in Low Dimension

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

Student t-distribution, 1 degree of freedom (red)
Gaussian (blue)



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

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

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

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

Gradient Interpretation

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

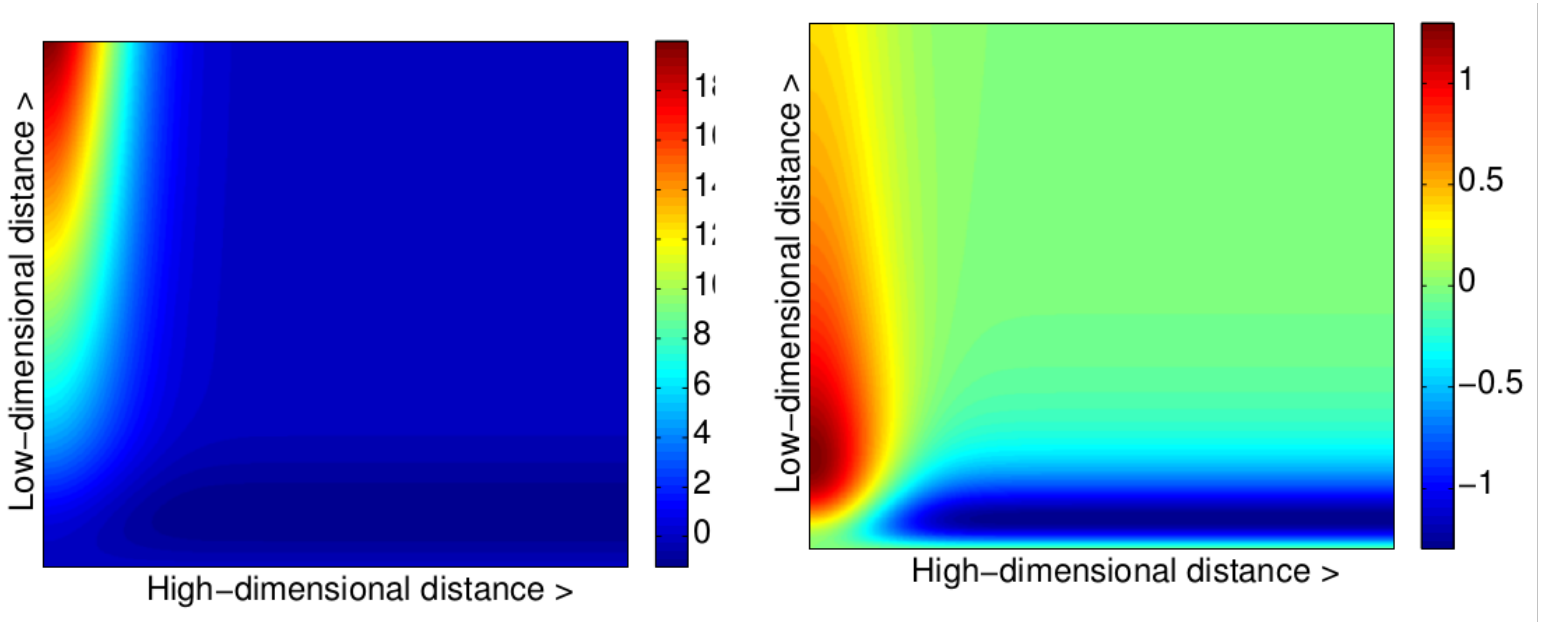
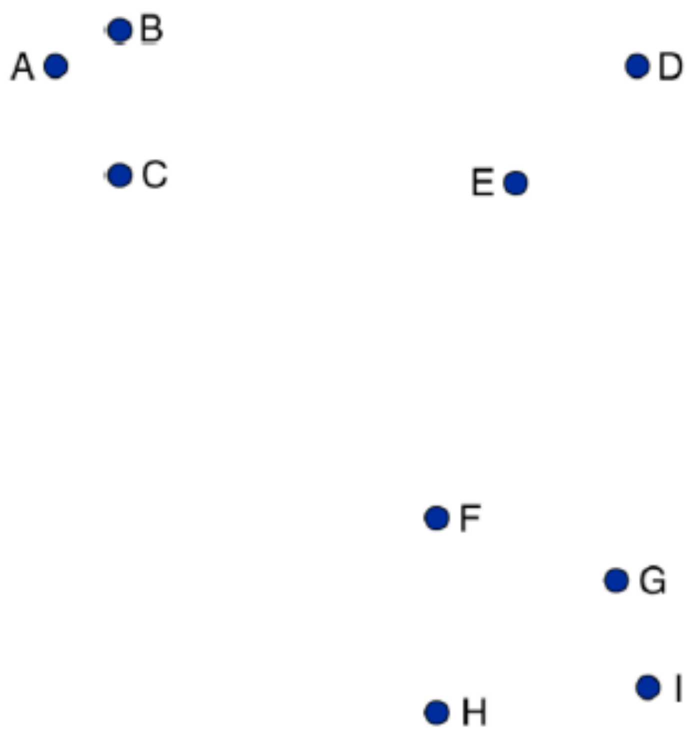


Figure : Gradient of SNE and t-SNE

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

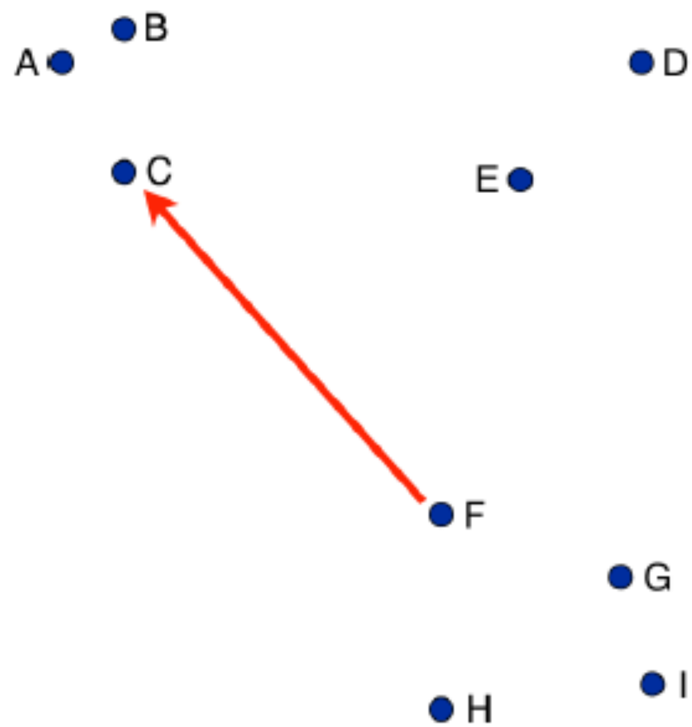
$$\frac{\partial C}{\partial y_i} = 4\sum_{j \neq i}(p_{ij} - q_{ij})(1 + ||y_i - y_j||^2)^{-1}(y_i - y_j)$$



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

- Displacement

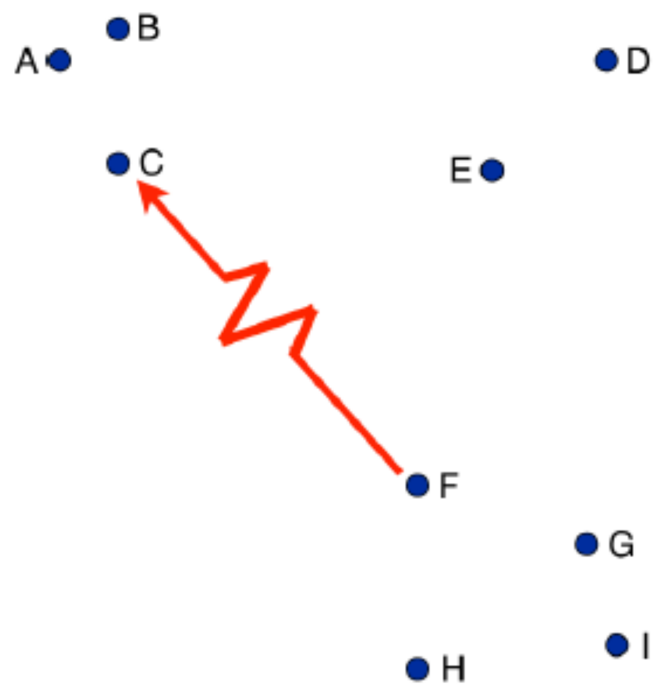
$$(y_i - y_j)$$



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

- Exertion / Compression

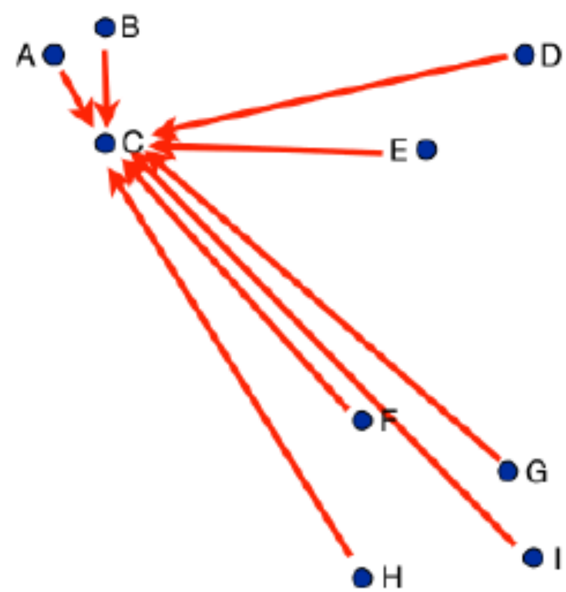
$$(p_{ij} - q_{ij})(1 + ||y_i - y_j||^2)^{-1}$$



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

- N-Body, summation

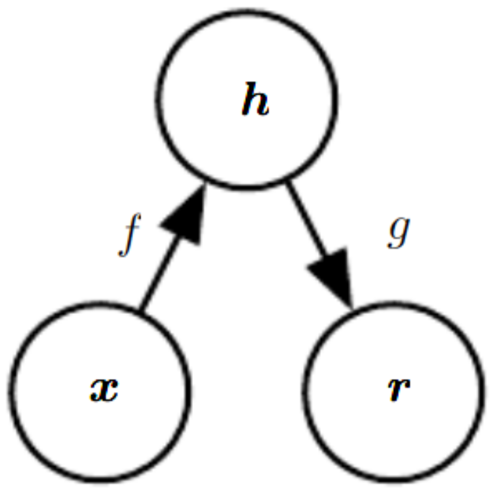
$$\frac{\partial C}{\partial y_i} = 4 \sum_{j \neq i} (p_{ij} - q_{ij})(1 + ||y_i - y_j||^2)^{-1}(y_i - y_j)$$



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

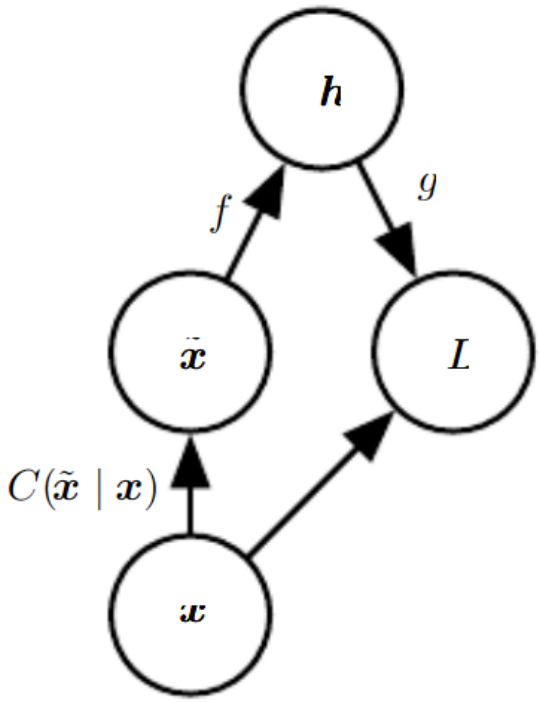
Autoencoders (Part III)

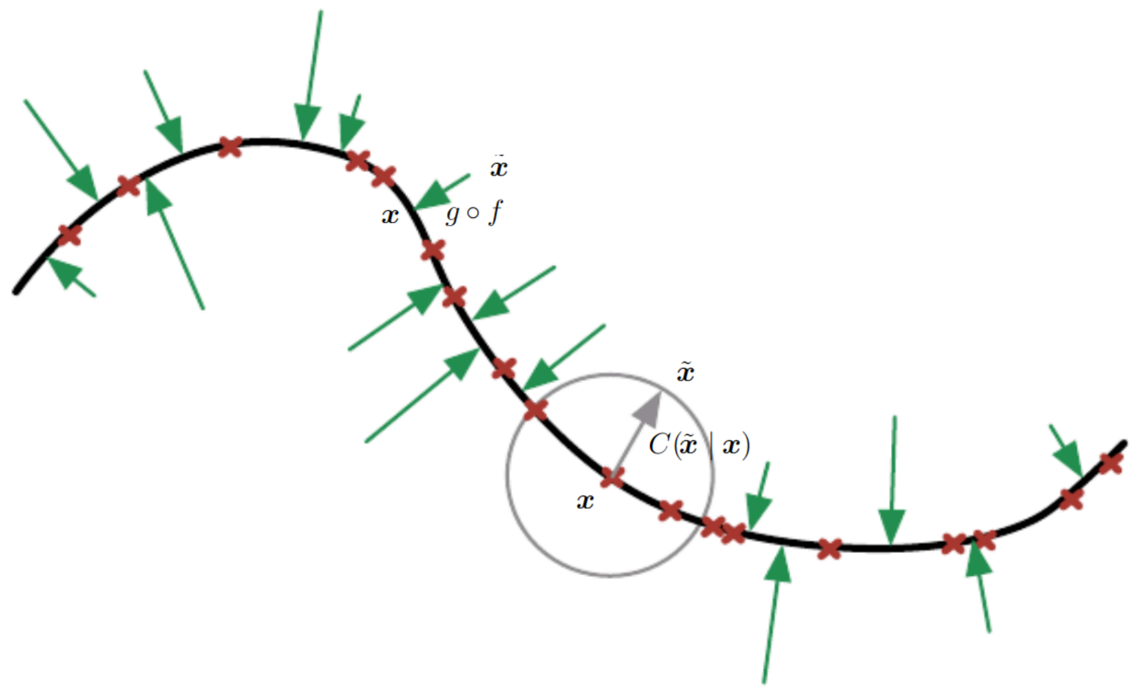
Autoencoders learn a latent representation for input data



$$L(\boldsymbol{x}, g(f(\boldsymbol{x})))$$

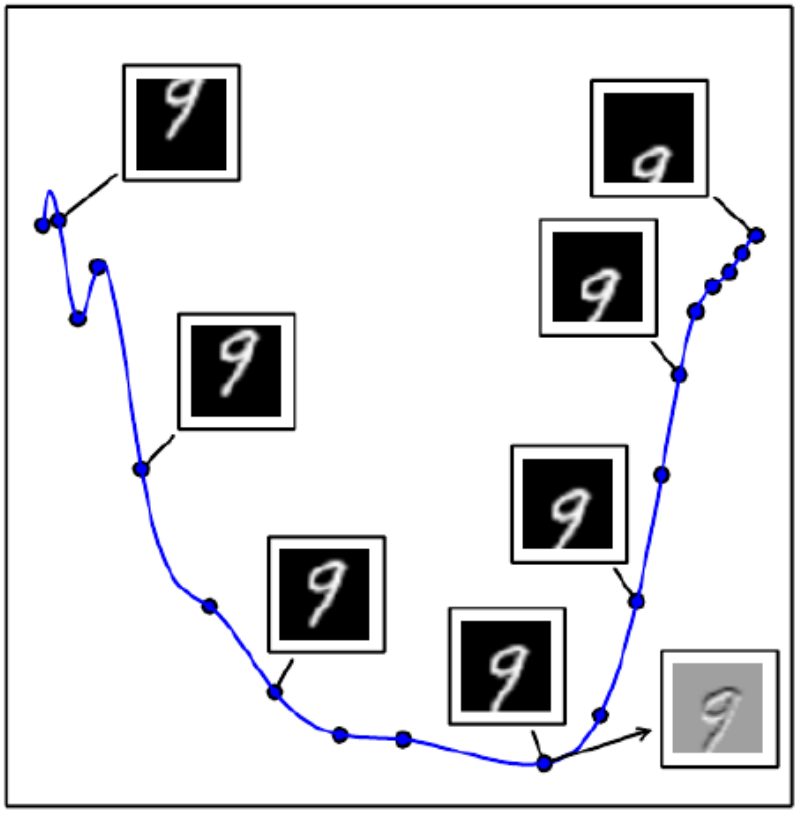
Denoising autoencoders recover signal corrupted by noise

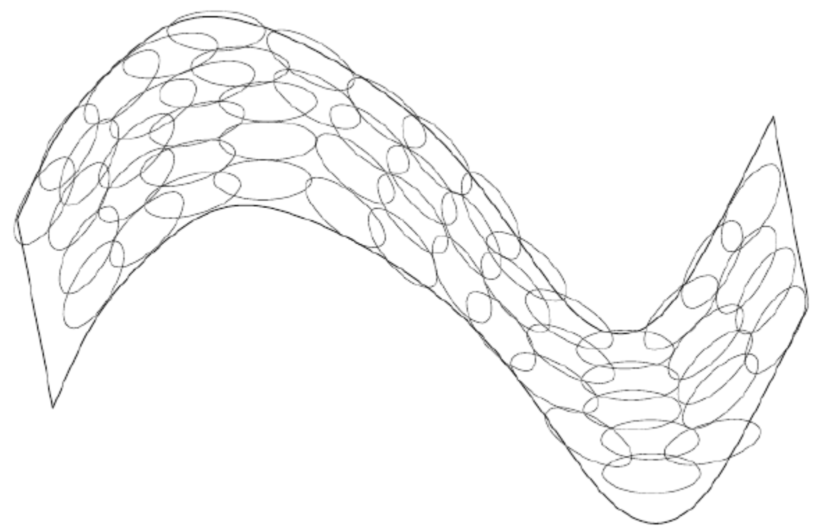




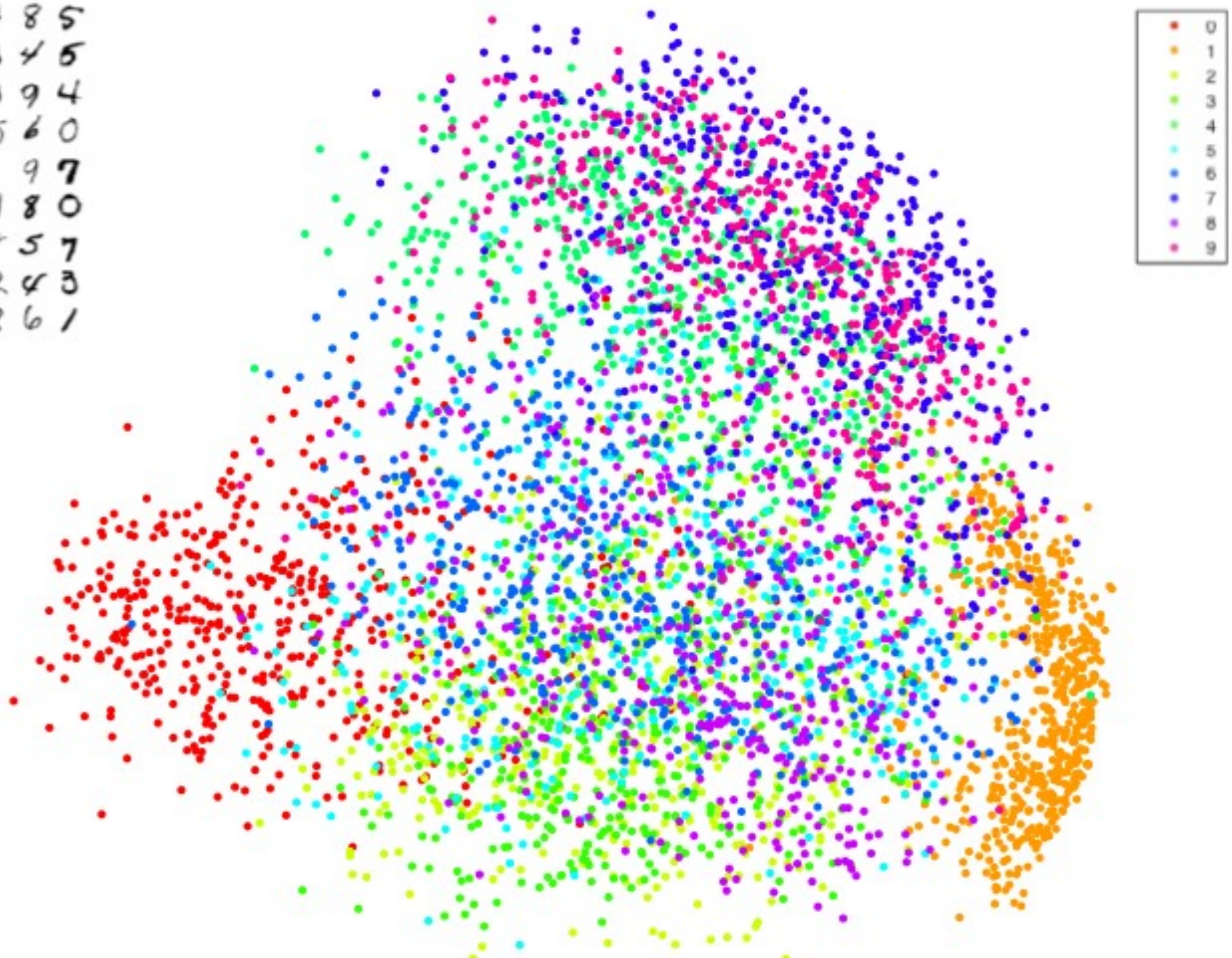
$$L(\boldsymbol{x}, g(f(\tilde{\boldsymbol{x}}))),$$

We can lean manifolds with autoencoders

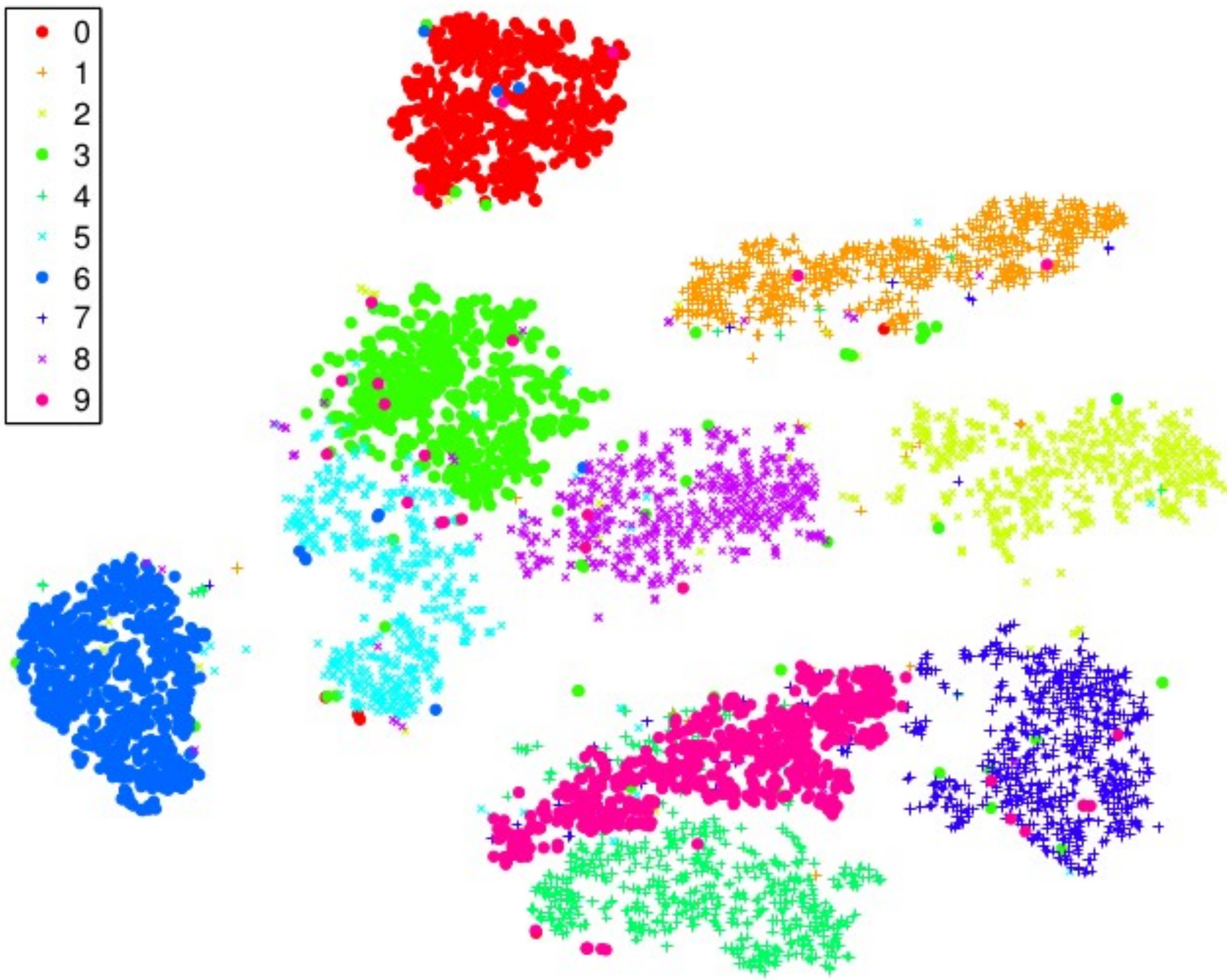




Principal Components Analysis



MNIST t-SNE



Cool interactive demos

- http://dpkingma.com/sgvb_mnist_demo/demo_old.html
- http://elf-project.sourceforge.net/autoencoder.html
- http://vdumoulin.github.io/morphing_faces/online_demo.html

FIN - Thank You

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