Unsupervised Dimensionality Reduction

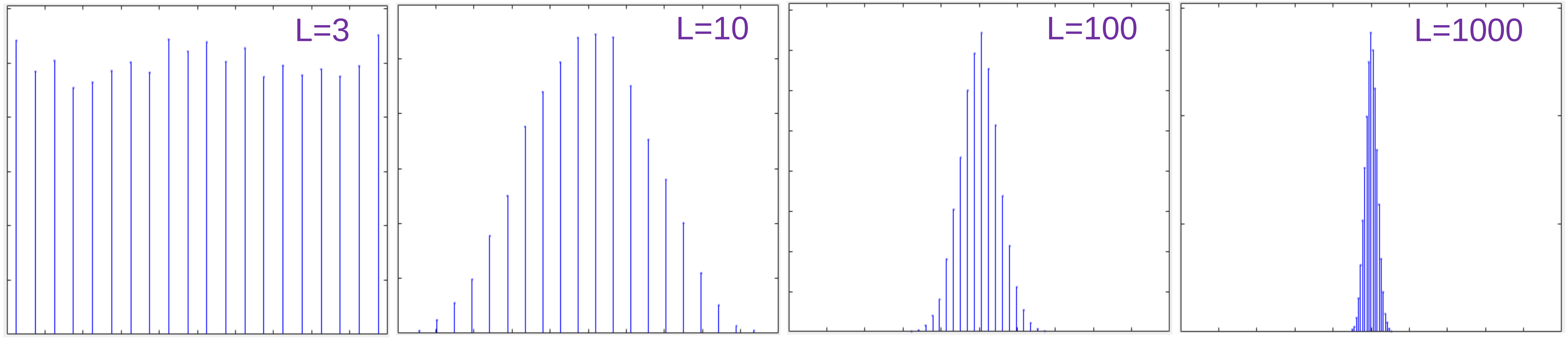
Dimensionality Reduction

- An important subfield of unsupervised learning is dimensionality reduction, which is to generate more compact (lower-dimensional) representations of samples than in their original space.
- Why dimensionality reduction?
  - Better efficiency (time and memory requirements)
  - Lower model complexity; less likely to overfit
  - Fewer data required for modeling data
- Dimensionality reduction leads to a lossy representation of the data, so it’s important to preserve as much information useful for the task as possible.

Curse of Dimensionality

(The is just an illustration of the concept.)

Assume that we have a unit sphere of **L** dimensions. The histograms of **cosine similarities** of random point pairs on the sphere are:



(horizontal axis: -1 ~ 1)

Implication: All point pairs are similarly (dis)similar for large **L**. ➔ Difficult to compare distances, group items, find neighbors, estimate local densities, etc. (Or we need # samples to grow exponentially with **L** to make these tasks reliable.)

Principle Component Analysis (PCA)

Motivation: When you have a large number of features for each sample, it is very likely that the features are correlated (and therefore somewhat redundant).

The objectives of PCA:

- Find a linear transform to map the feature vectors to a lower-dimensional space.

$$\boldsymbol{y} = A^T \boldsymbol{x}$$

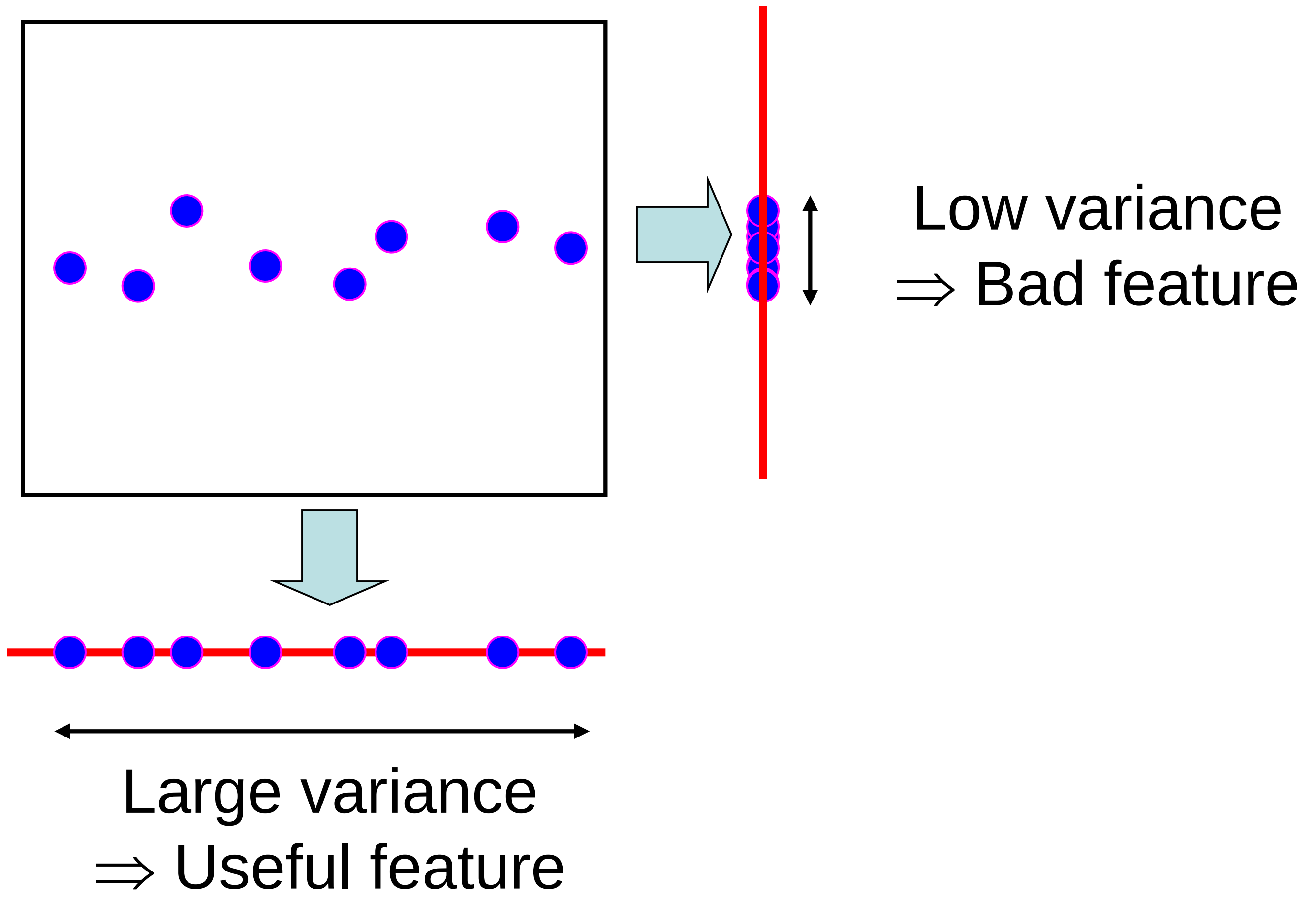
- Make the components in the transformed space uncorrelated. ➔ Use the eigenvectors of the covariance matrix as the basis vectors of the linear transform.

$$E[y_i y_j] = 0, \quad \forall i \neq j$$

- Retain as much variance (a measure of information content) as possible after the transform. ➔ Keep the eigenvectors associated with the larges l eigenvalues (l = dimension of target space.)

Variance and Information Content

Variance is usually used as a measure of information contained in data. For example:



PCA as Data Approximation

This is the cost function of PCA, which is the total squared errors of reconstruction (projection of the transformed points into the original space).

$$J = \sum_{i=1}^{N} \left\| \boldsymbol{x}_i - \sum_{j=1}^{l} (\boldsymbol{x}_i^T \boldsymbol{e}_j)\boldsymbol{e}_j \right\|^2$$

with the constraints

$$\boldsymbol{e}_i{}^T \boldsymbol{e}_j = 0 \quad \text{for } \forall i \neq j$$

$$\boldsymbol{e}_i{}^T \boldsymbol{e}_i = 1 \quad \text{for } \forall i$$

Here $\boldsymbol{e}_1$, $\boldsymbol{e}_2$, …, $\boldsymbol{e}_l$ are the l orthonormal basis vectors of the subspace.

It can be proved that this cost function, when minimized, correspond to the total variance along the discarded eigenvectors (dimensions).

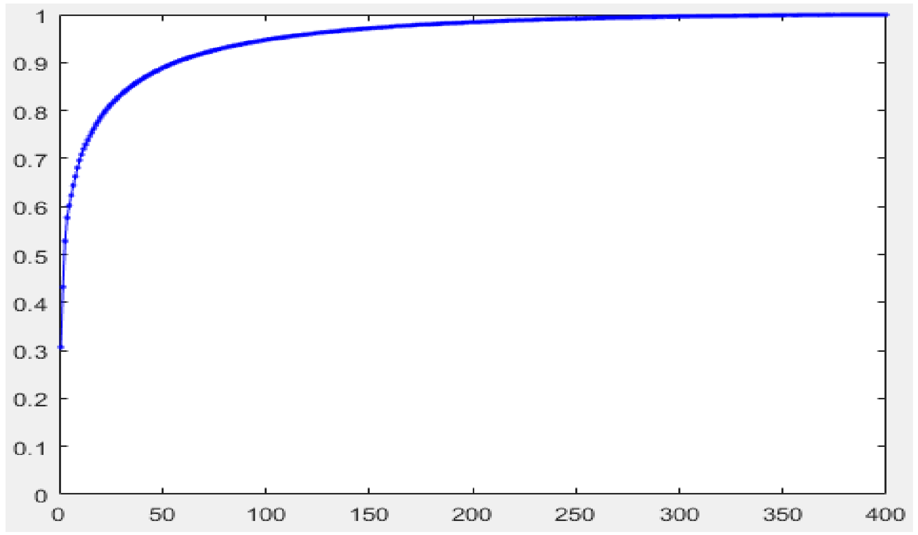
Example PCA Application: Eigenfaces

An example application of PCA is in face recognition, where each input vector has the dimension of the number of pixels in a face image. A popular technique is "eigenface", where the most important eigenvectors obtained by PCA are used as the basis vectors. The recognition is then based on the projection of an input image to these "eigenfaces".

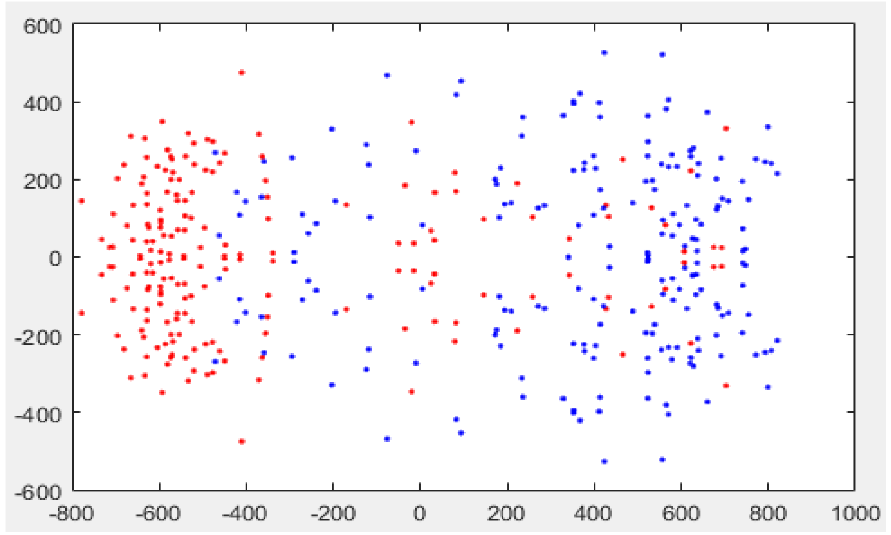
200 images
(100 males + 100 females)
+ horizontally flipped images
size 40x40
➔ 400 1600-D vectors

Example PCA Application: Eigenfaces

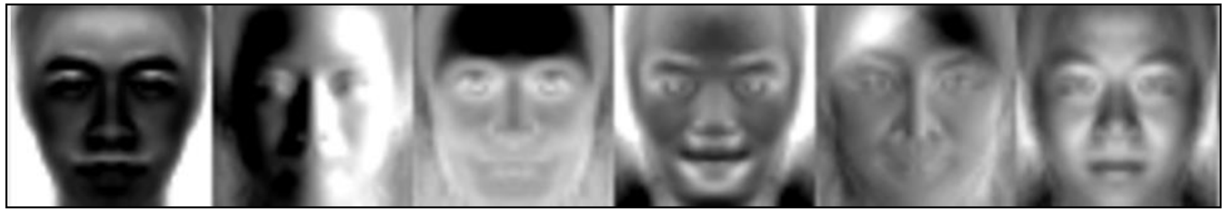
Accumulated variances (normalized):



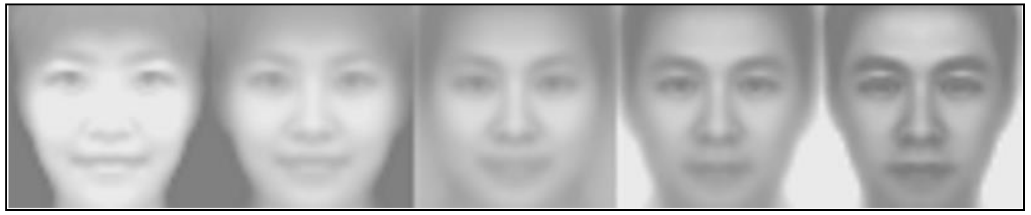
Projections in the first two PCA dimensions:



The first few eigenfaces:

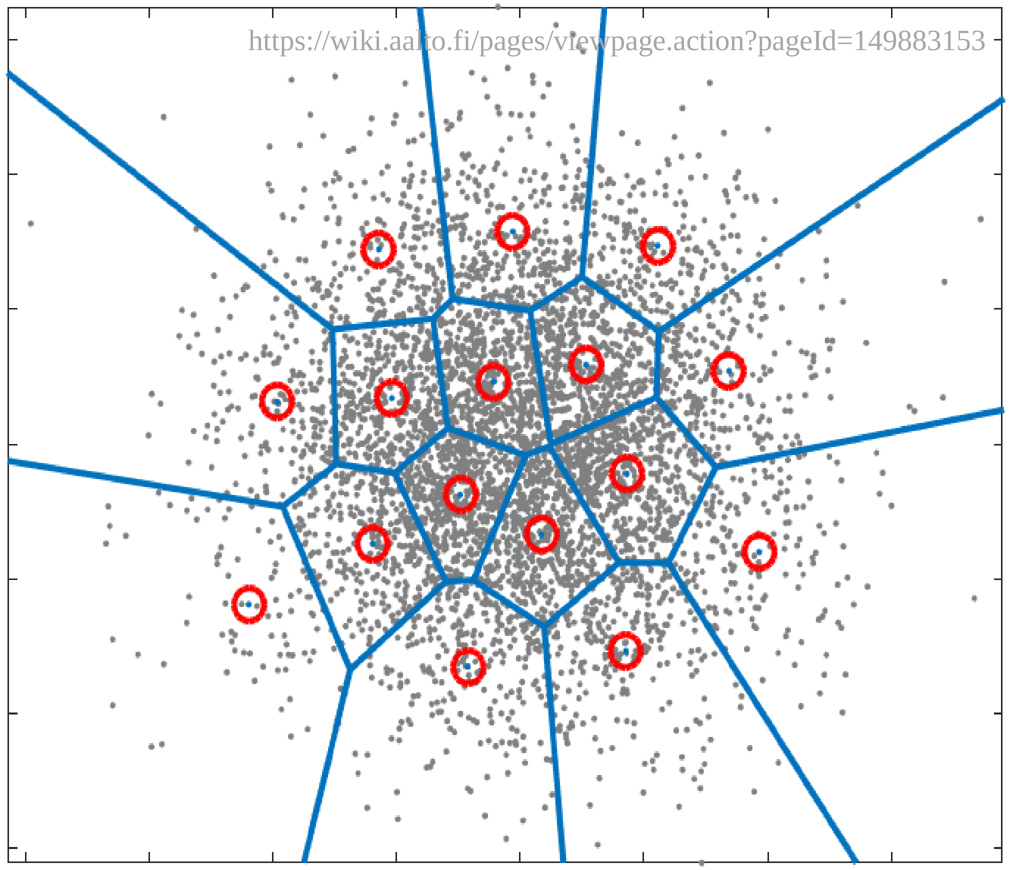


Reconstructed faces along the first PCA dimension:



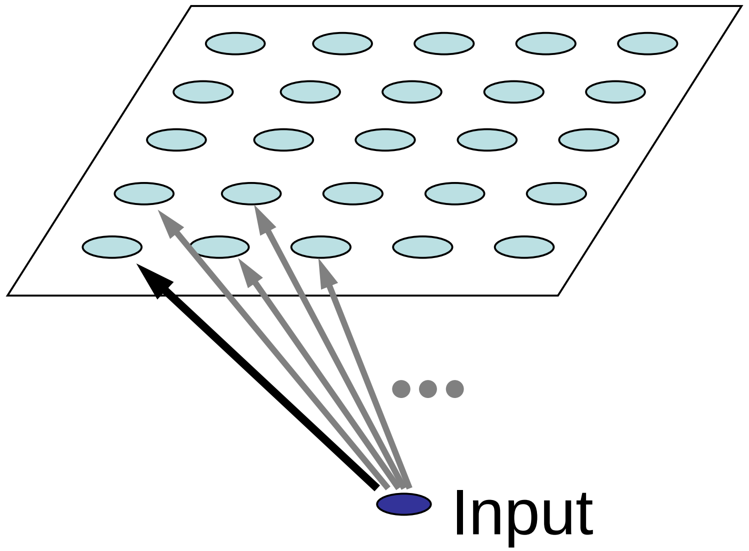
Vector Quantization (VQ)

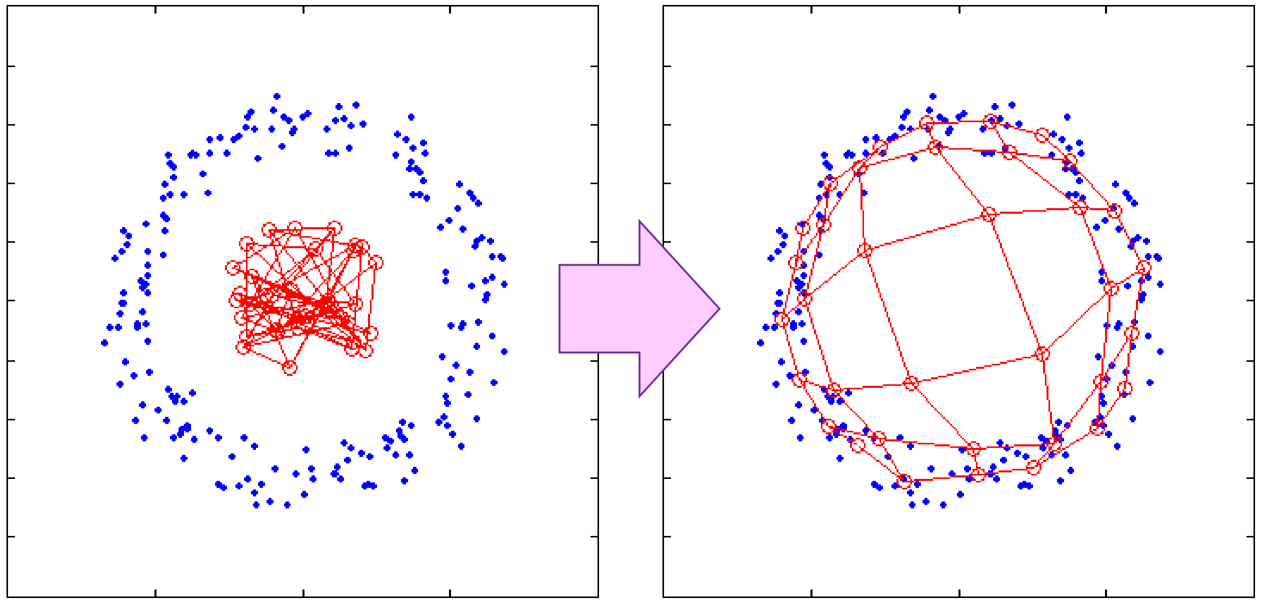
- VQ uses the same cost function as k-means; the prototypes become the code vectors.
- K-means uses batch closed-form update, while VQ uses online gradient-descent update.
- The purpose is not to find cluster structures, but to code a large number of samples more compactly.
- One application of VQ is lossy compression. (It's a part of many earlier image codecs.)



Self-Organizing Maps (SOM)

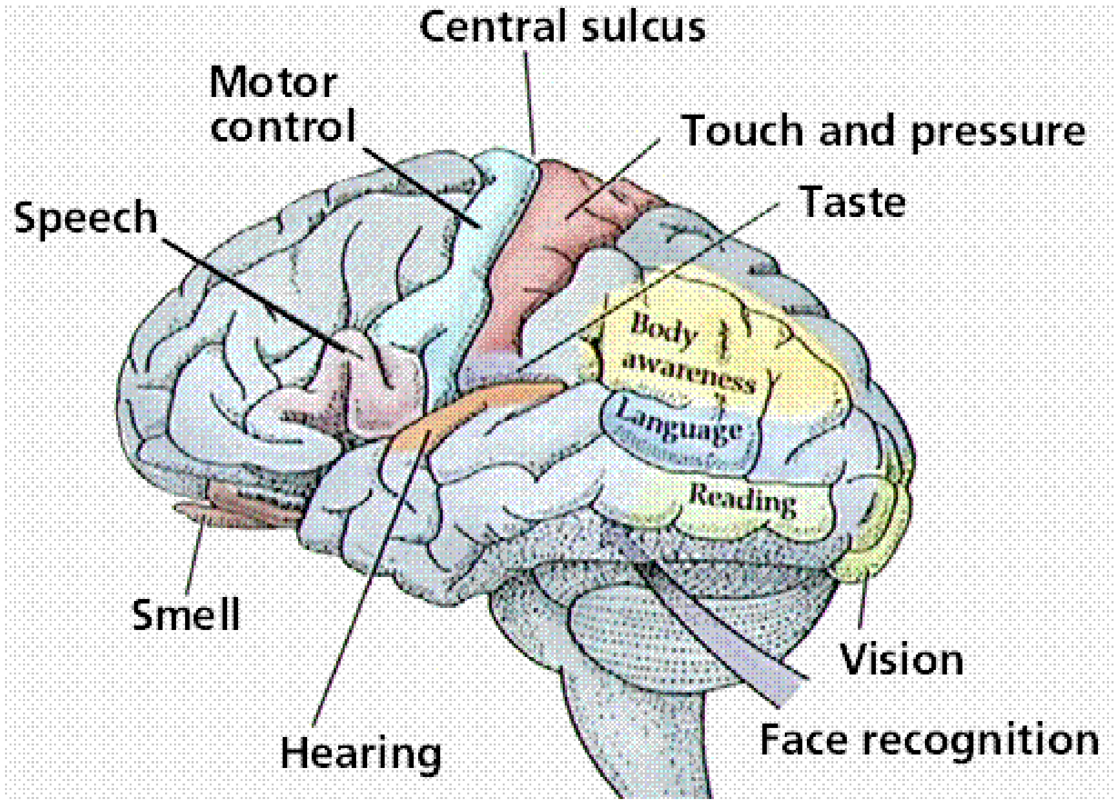
- A SOM has a single layer of neurons arranged on a lattice (2-D is most common).
- Each neuron represents a point in the feature space.
- Nearby neurons on the lattice should respond to inputs similarly.
- VQ with neuron cooperation ("dragging-along") during training.
- The resulting lattice gives an organized representation of the inputs.

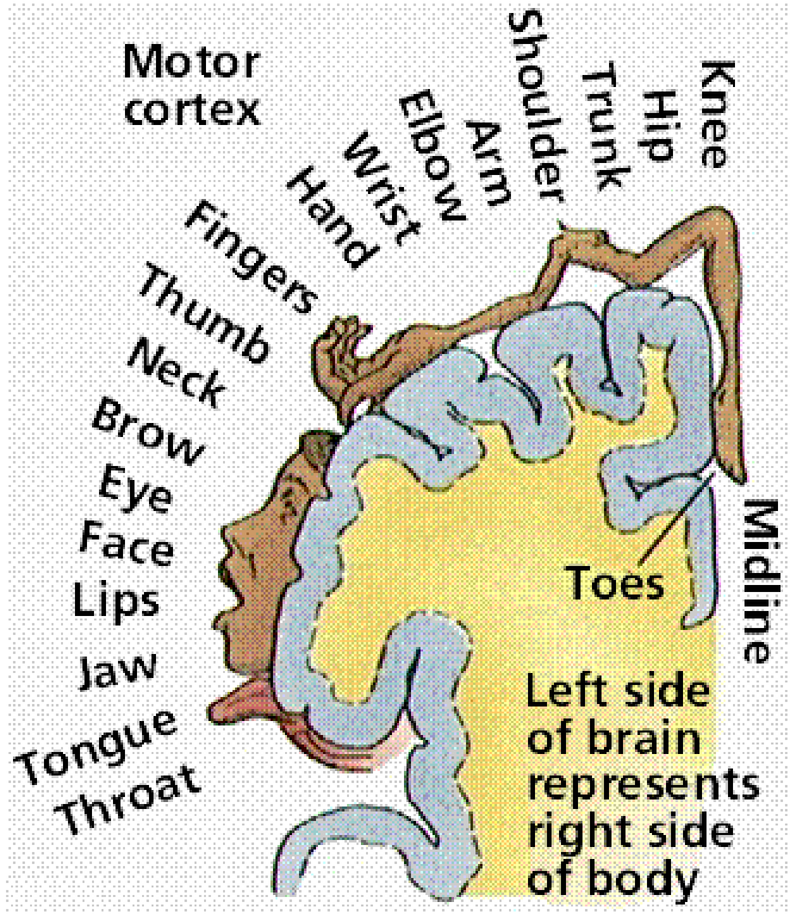




Biological Origin of SOM

SOM is inspired by how the information processing in a human brain is distributed and organized at the cerebral cortex:



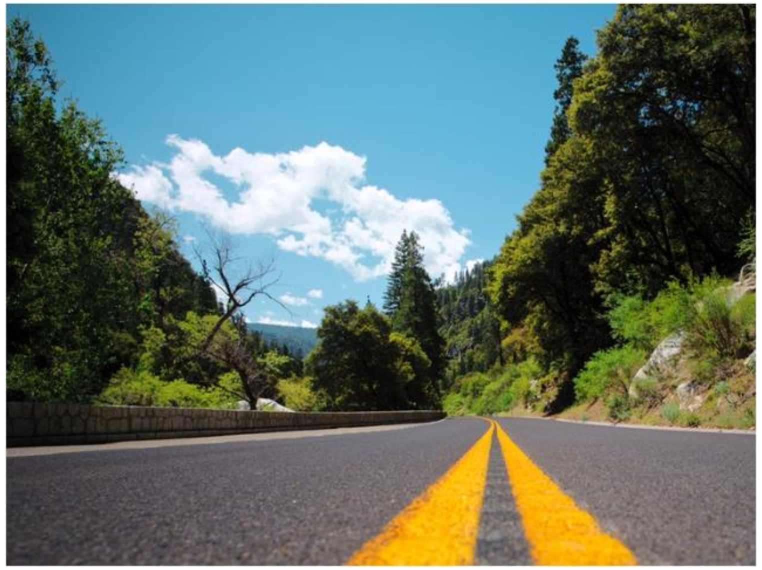


Applications of SOM

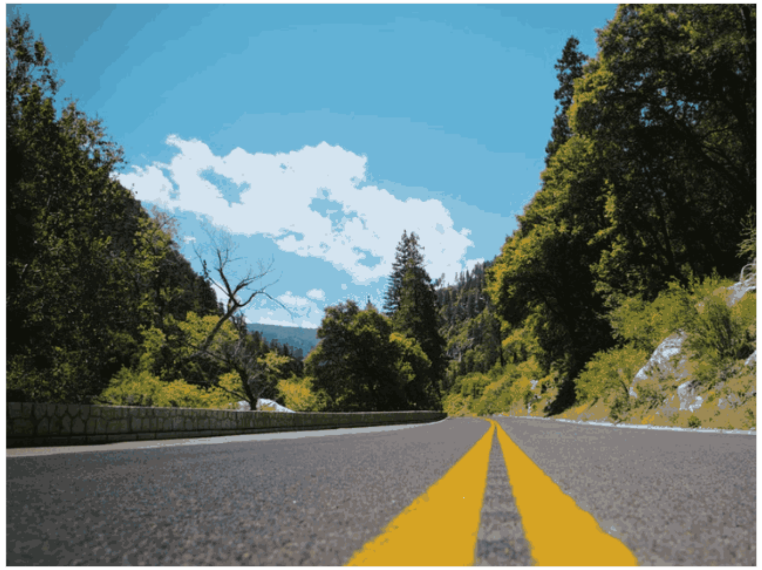
- A reduced-dimension representation of the original data by projecting input samples to the space of the weight vectors.
- The distribution of neuron weight vectors approximates the "density" of the training data.
- The approximate topology (proximity) is preserved. (This is how SOM differs from methods like VQ.)
- Many tasks (such as training a classifier) can be done on this lower dimensional representation. As a result, SOM is often called **SOFM** (self-organizing feature map) as well.
- Visualization of high-dimensional data.

Applications of SOM: Example

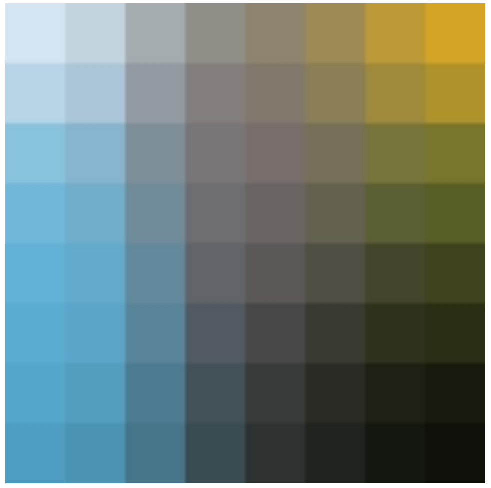
- SOM applied to the RGB values of image pixels.



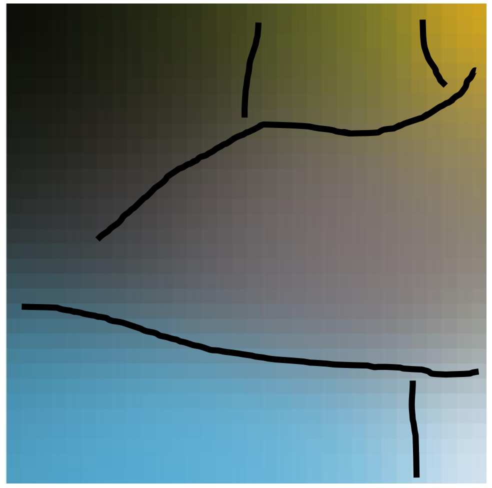
64 colors



8x8 SOFM



32x32 SOFM



Embedding

We will start to encounter the term "**embedding**"… What does it mean?

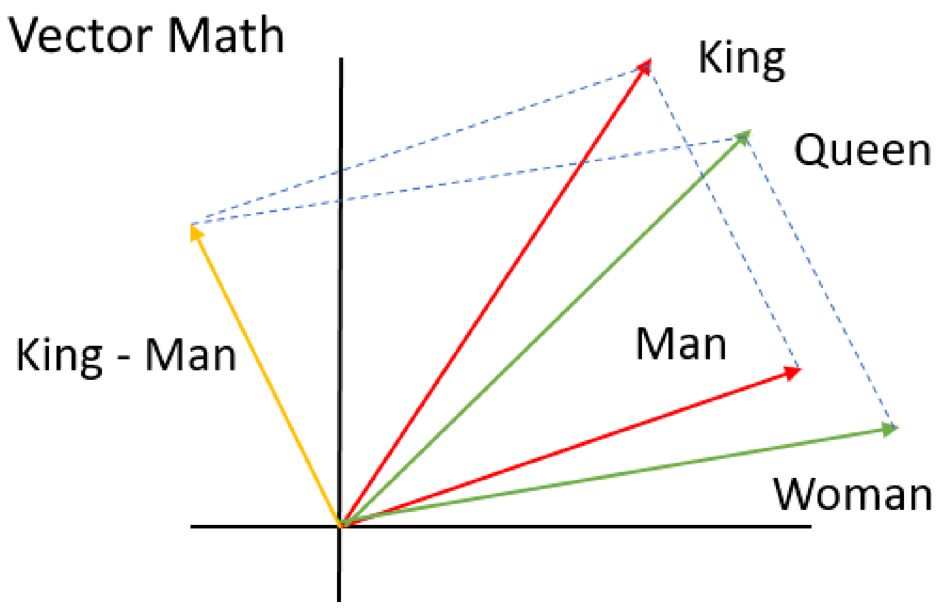
- "*A map which maps a subspace (smaller structure) to the whole space (larger structure).*" (Wikipedia)
- "A relatively low-dimensional space into which you can translate high-dimensional vectors." (Google Machine Learning Crash Course)
- "Representations of values or objects like text, images, and audio that are designed to be consumed by machine learning models and semantic search algorithms." (Cloudflare)

The concepts of embedding and dimensionality reduction are highly related but not exactly the same. We can somehow think of embeddings as vector representations of complex data that make them easier to process.

Word2Vec: Word Embedding

- Embedding of text tokens (words) is the most commonly mentioned example.
- In natural language processing, the space of words has very high dimensionality with one-hot coding.
  - Inefficient processing
  - Relations between words missed (no context in the codes)
- A lower dimensional embedding is desired where the dimensions can capture meanings of words.

An example in English:



https://medium.com/analytics-vidhya/word-embeddings-in-nlp-word2vec-glove-fasttext-24d4d4286a73

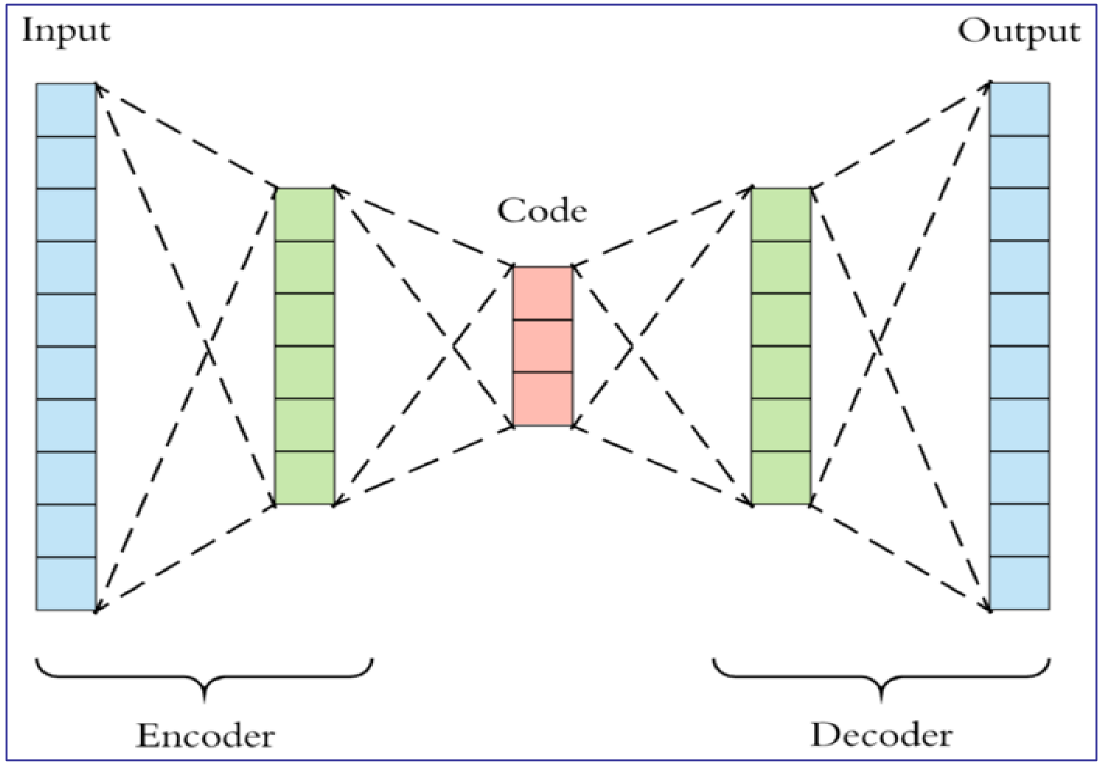
Learning Word Embedding

- But how do we get the word embedding (the vector space of the embedding, and the vectors presenting individual words)?
- We have to learn from the context (in practice, this means the surrounding words):

  "*A word is characterized by the company it keeps.*" (Firth, 1950s)
- Train with (central-word, context-words) pairs from a large corpus.
- We will leave more detail of the topic to later lectures…

Autoencoder

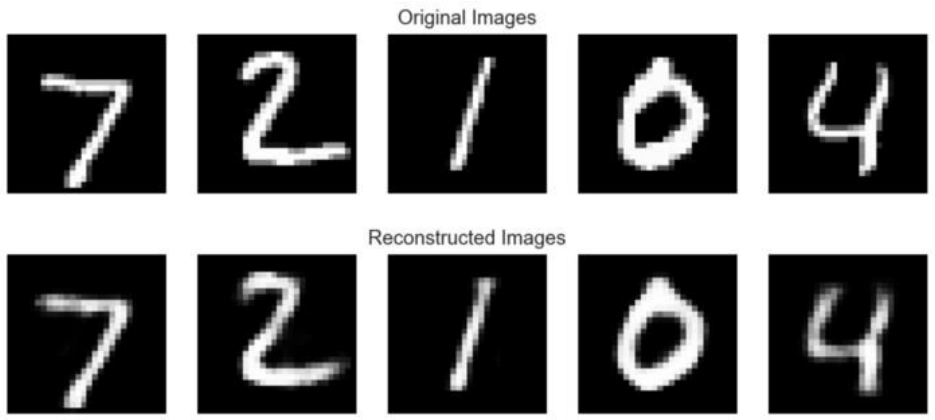
- **Autoencoder** is a learning based technique for dimensionality reduction.
- The objective is for the output to “reproduce” the input. (This is considered **self-supervised learning**, which we will talk about in later lectures.)
- The "information bottleneck" leads to dimensionality reduction, forcing the network to learn only the “important” information from the training samples.
- The "code" part can be considered the "embedding" of the input.



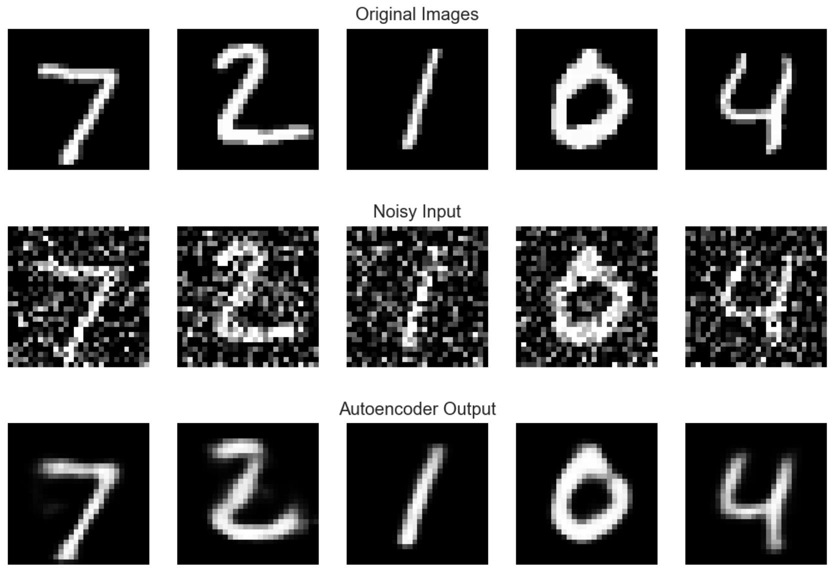
https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798

Autoencoder Examples

- Reconstruction from a 32-element code. (Original input space has 784 dimensions.



- Image denoising. (Trained with noisy-clean image pairs.)

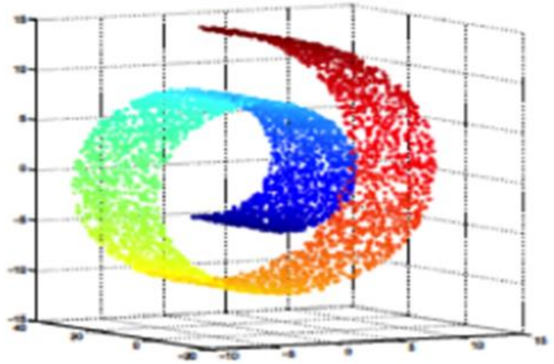


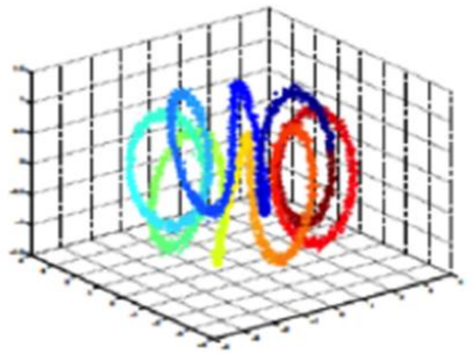
- Many other applications …

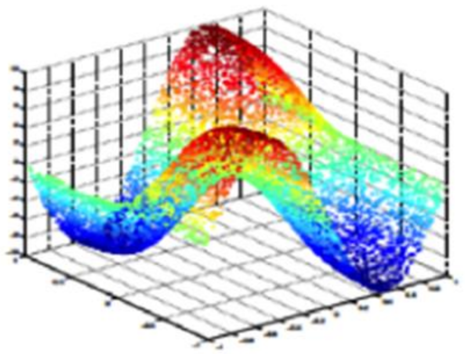
https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798

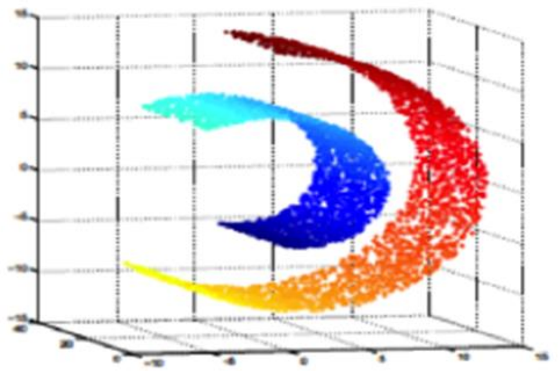
Nonlinear Embedding and Manifolds

- **Manifolds**: Low dimensional structures in a high dimensional space.
- A few synthetic examples are shown here.
- A 1-D manifold is also called a **principal curve**. 1-D SOM is a technique that can be used to find principal curves.
- It is assumed that, for many complex data (e.g., face images of a person), they are actually more like manifolds in a high-dimensional space.
- We can imagine the embedding as the "flattening" of the manifold in a lower-dimensional space.


(a) Swiss roll dataset.


(b) Helix dataset.


(c) Twinpeaks dataset.

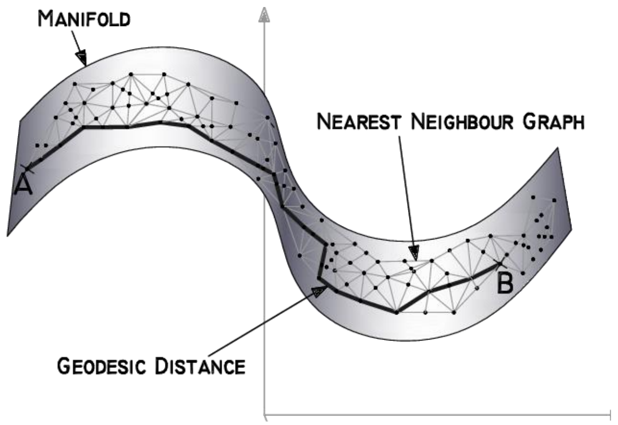

(d) Broken Swiss roll dataset.

Example Methods of Low-Dimensional Embedding

- Multi-Dimensional Scaling (MDS): A low-dimensional mapping that preserves the pairwise distances between the data points. A typical objective function is

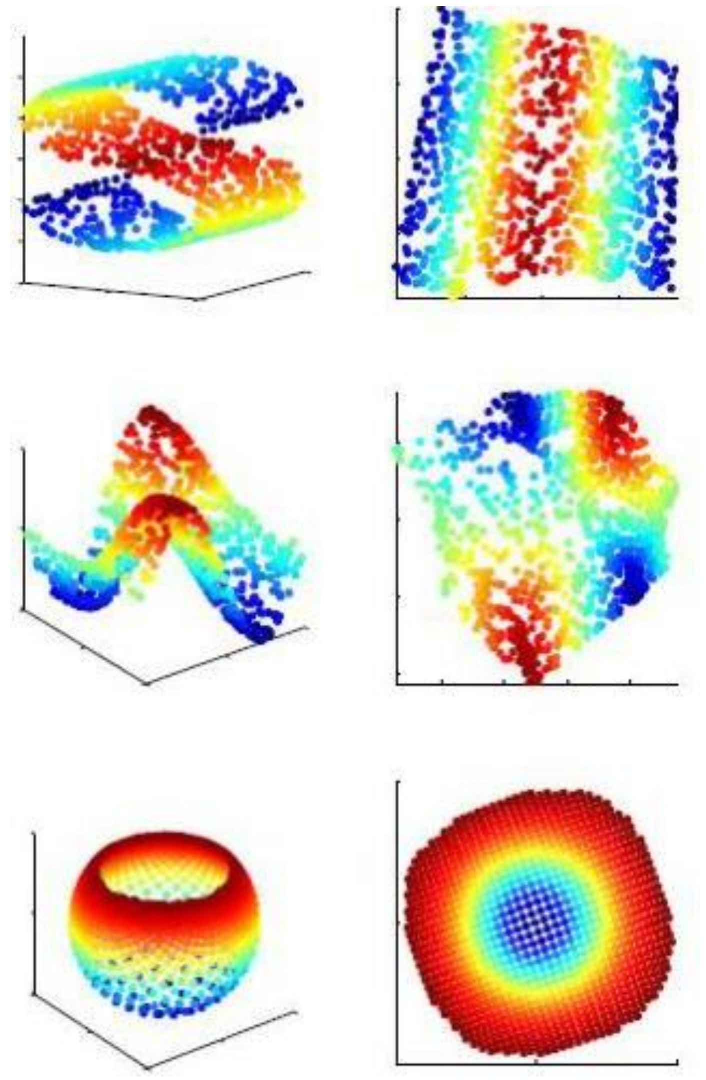
$$Stress = \sqrt{\sum_{i \neq j} \left(d_{ij} - \| \mathbf{z}_i - \mathbf{z}_j \| \right)^2}$$

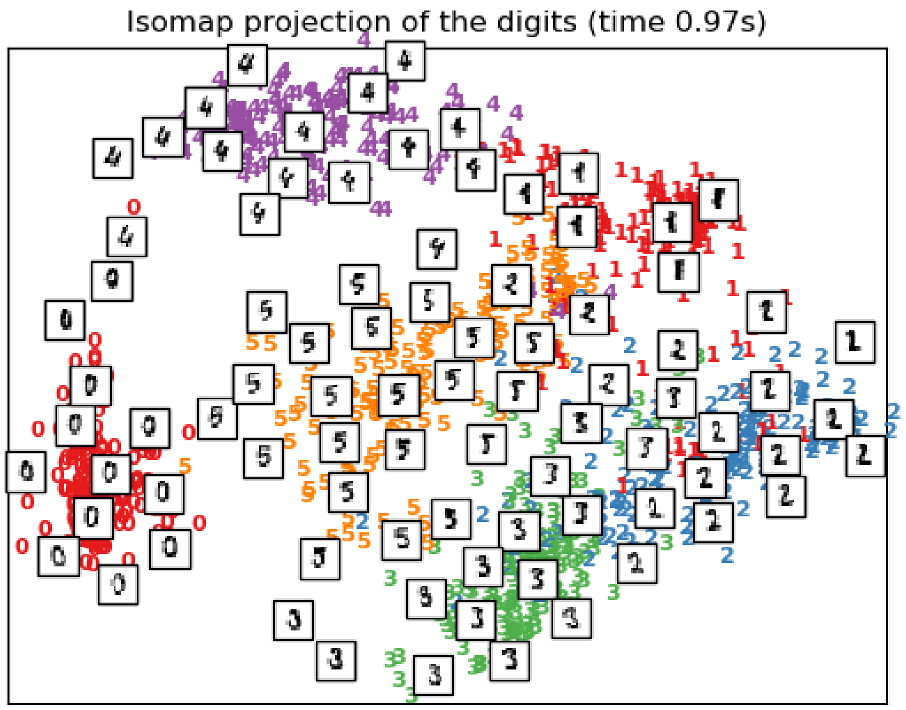
- Isometric Mapping (ISOMAP): A low-dimensional mapping that preserves the pairwise geodesic distances between data points.



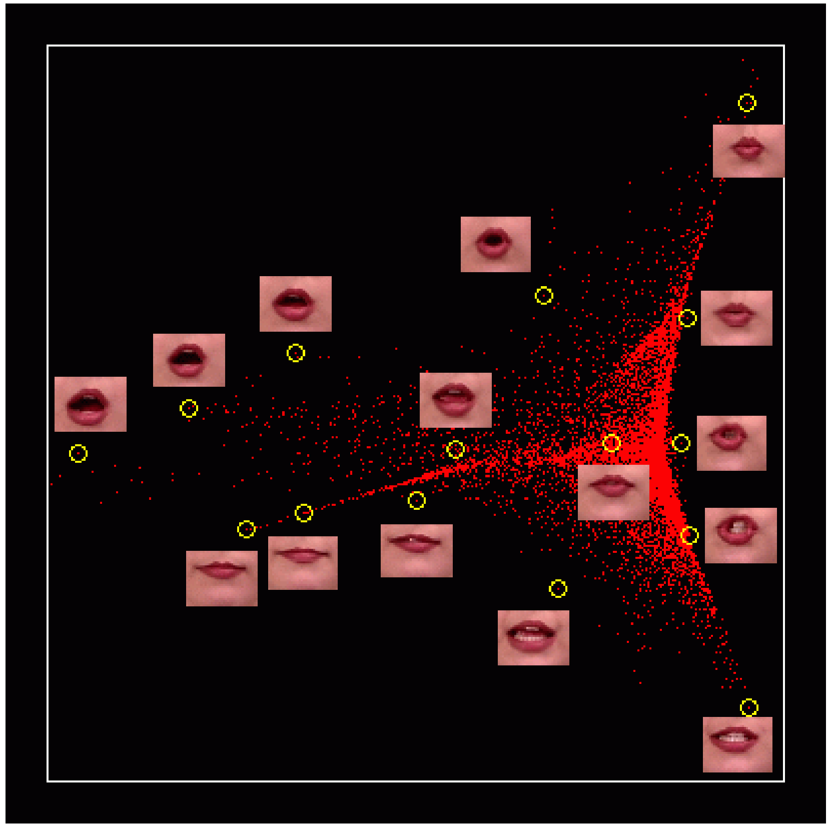
- Locally Linear Embedding (LLE): The objective is to preserve local linearity (approximating data points as similar linear combinations of their respective neighbors before and after projection).

Manifold Learning / Embedding Examples





https://scikit-learn.org/0.22/auto_examples/manifold/plot_lle_digits.html



https://cs.nyu.edu/~roweis/lle/lips.html

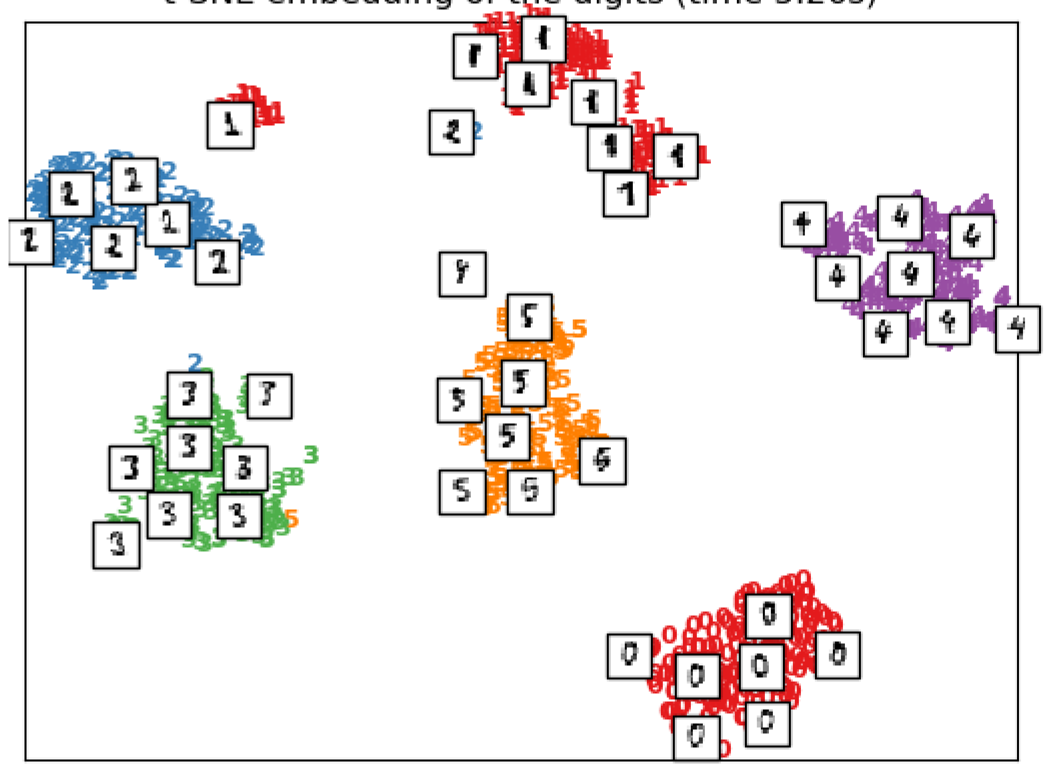
t-SNE

t-Distributed Stochastic Neighbor Embedding (t-SNE):

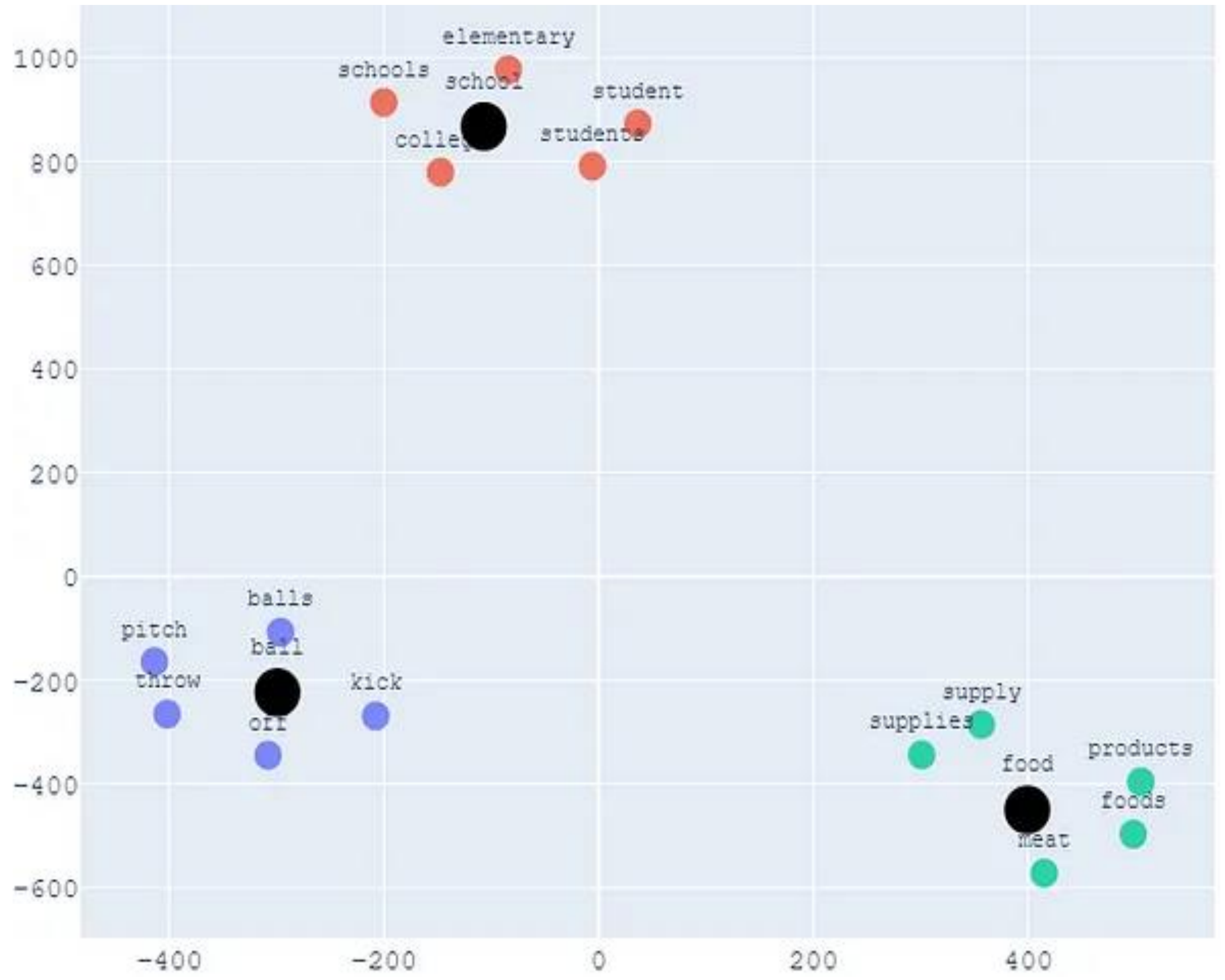
- The objective is that the local conditional probabilities (pairwise between data points) in a lower dimensional space is matched to the local conditional probabilities in the original space.
- The pdf: normal distribution in original space, t-distribution in lower dimensional space.
- The arrangements of the points in the low-dimensional space are initialized randomly and updated with gradient descent.
- Perplexity: A hyper-parameter that controls the effective number of neighbors for each data point by making the local distribution bandwidths adaptive to local densities.

t-SNE Examples

t-SNE embedding of the digits (time 5.26s)



https://scikit-learn.org/0.22/auto_examples/manifold/plot_lle_digits.html



https://towardsdatascience.com/visualizing-word-embedding-with-pca-and-t-sne-961a692509f5