Reducing Data Dimension

Recommended reading:

- Bishop, chapter 3.6, 8.6
- Wall et al., 2003

Machine Learning 10-701
April 2005

Tom M. Mitchell
Carnegie Mellon University

Outline

- Feature selection
  - Single feature scoring criteria
  - Search strategies

- Unsupervised dimension reduction using all features
  - Principle Components Analysis
  - Singular Value Decomposition
  - Independent components analysis

- Supervised dimension reduction
  - Fisher Linear Discriminant
  - Hidden layers of Neural Networks

Dimensionality Reduction

Why?

- Learning a target function from data where some features are irrelevant
- Wish to visualize high dimensional data
- Sometimes have data whose “intrinsic” dimensionality is smaller than the number of features used to describe it - recover intrinsic dimension

Supervised Feature Selection

Supervised Feature Selection

Problem: Wish to learn f: X → Y, where X=<X_1, …X_N>

But suspect not all X_i are relevant

Approach: Preprocess data to select only a subset of the X_i

- Score each feature, or subsets of features
  - How?
- Search for useful subset of features to represent data
  - How?

Scoring Individual Features X_i

Common scoring methods:

- Training or cross-validated accuracy of single-feature classifiers f_i: $X_i \rightarrow Y$
- Estimated mutual information between X_i and Y :

$$\hat{I}(X_i, Y) = \sum_k \sum_y \hat{P}(X_i = k, Y = y) \log \frac{\hat{P}(X_i = k, Y = y)}{\hat{P}(X_i = k)\hat{P}(Y = y)}$$

- χ^2 statistic to measure independence between X_i and Y
- Domain specific criteria
  - Text: Score "stop" words ("the", "of", …) as zero
  - fMRI: Score voxel by T-test for activation versus rest condition
  - …

Choosing Set of Features

Common methods:

Forward1: Choose the n features with the highest scores

Forward2:

- Choose single highest scoring feature X_k
- Rescore all features, conditioned on X_k being selected
  - E.g, Score(X_i)= Accuracy($\{X_i, X_k\}$)
  - E.g., Score(X_i) = $I(X_i, Y \mid X_k)$
- Repeat, calculating new conditioned scores on each iteration

Choosing Set of Features

Common methods:

Backward1: Start with all features, delete the n with lowest scores

Backward2: Start with all features, score each feature conditioned on assumption that all others are included. Then:

- Remove feature with the lowest (conditioned) score
- Rescore all features, conditioned on the new, reduced feature set
- Repeat

Feature Selection: Text Classification

Approximately 10^5 words in English

[Rogati&Yang, 2002]

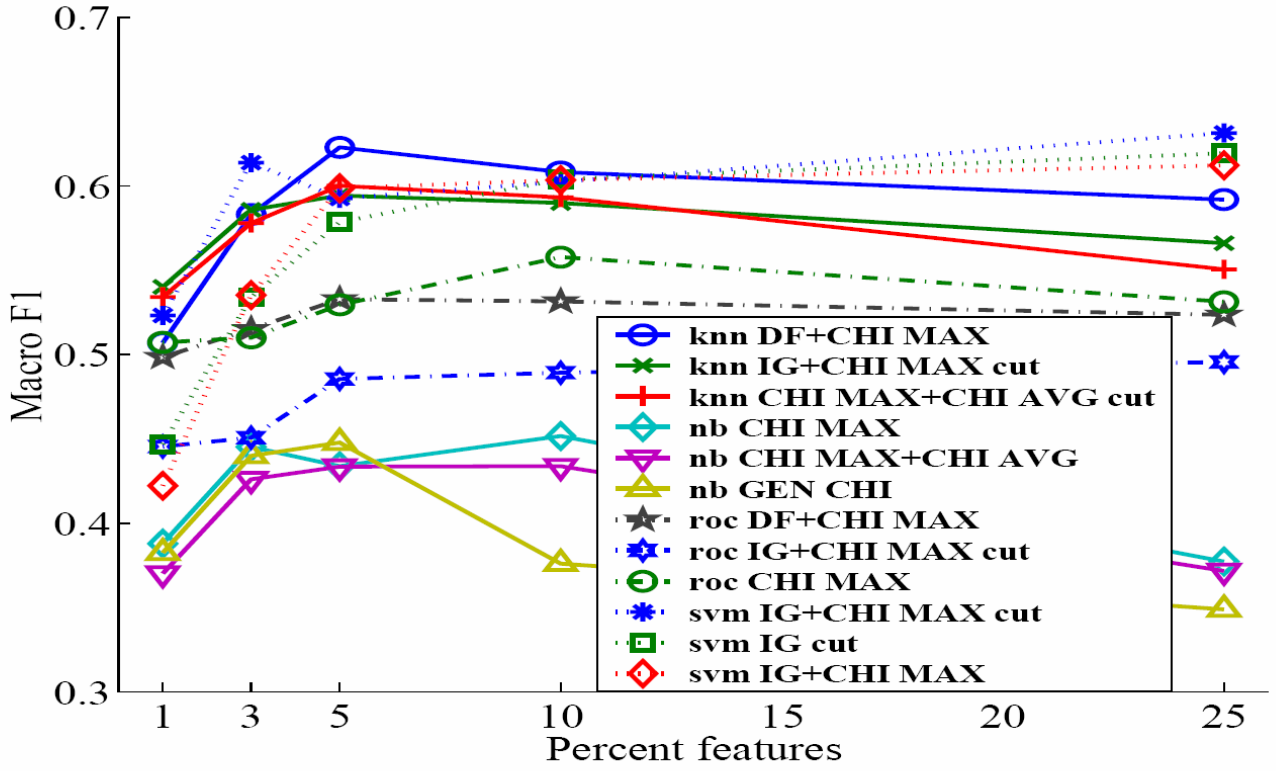


Figure 2: Top 3 feature selection methods for Reuters-21578 (Macro F1)

IG=information gain, chi= χ^2 , DF=doc frequency,

Impact of Feature Selection on Classification of fMRI Data

[Pereira et al., 2005]

Accuracy classifying category of word read by subject

#voxels	mean	subjects 233B	329B	332B	424B	474B	496B	77B	86B
50	0.735	0.783	0.817	0.55	0.783	0.75	0.8	0.65	0.75
100	0.742	0.767	0.8	0.533	0.817	0.85	0.783	0.6	0.783
200	0.737	0.783	0.783	0.517	0.817	0.883	0.75	0.583	0.783
300	**0.75**	**0.8**	**0.817**	**0.567**	**0.833**	**0.883**	**0.75**	**0.583**	**0.767**
400	0.742	0.8	0.783	0.583	0.85	0.833	0.75	0.583	0.75
800	0.735	0.833	0.817	0.567	0.833	0.833	0.7	0.55	0.75
1600	0.698	0.8	0.817	0.45	0.783	0.833	0.633	0.5	0.75
all (~2500)	0.638	0.767	0.767	0.25	0.75	0.833	0.567	0.433	0.733

Table 1: **Average accuracy across all pairs of categories, restricting the procedure to use a certain number of voxels for each subject.** The highlighted line corresponds to the best mean accuracy, obtained using 300 voxels.

Voxels scored by p-value of regression to predict voxel value from the task

Summary: Supervised Feature Selection

Approach: Preprocess data to select only a subset of the X_i

- Score each feature
  - Mutual information, prediction accuracy, …
- Find useful subset of features based on their scores
  - Greedy addition of features to pool
  - Greedy deletion of features from pool
  - Considered independently, or in context of other selected features

Always do feature selection using training set only (not test set!)

- Often use nested cross-validation loop:
  - Outer loop to get unbiased estimate of final classifier accuracy
  - Inner loop to test the impact of selecting features

Unsupervised Dimensionality Reduction

Unsupervised mapping to lower dimension

Differs from feature selection in two ways:

- Instead of choosing subset of features, create new features (dimensions) defined as functions over all features

- Don’t consider class labels, just the data points

Principle Components Analysis

- Idea:
  - Given data points in d-dimensional space, project into lower dimensional space while preserving as much information as possible
    - E.g., find best planar approximation to 3D data
    - E.g., find best planar approximation to 10^4 D data
  - In particular, choose projection that minimizes the squared error in reconstructing original data

PCA: Find Projections to Minimize Reconstruction Error

Assume data is set of d-dimensional vectors, where nth vector is

$$\mathbf{x}^n = \langle x_1^n \ldots x_d^n \rangle$$

We can represent these in terms of any d orthogonal basis vectors

$$\mathbf{x}^n = \sum_{i=1}^{d} z_i^n \mathbf{u}_i; \quad \mathbf{u}_i^T \mathbf{u}_j = \delta_{ij}$$

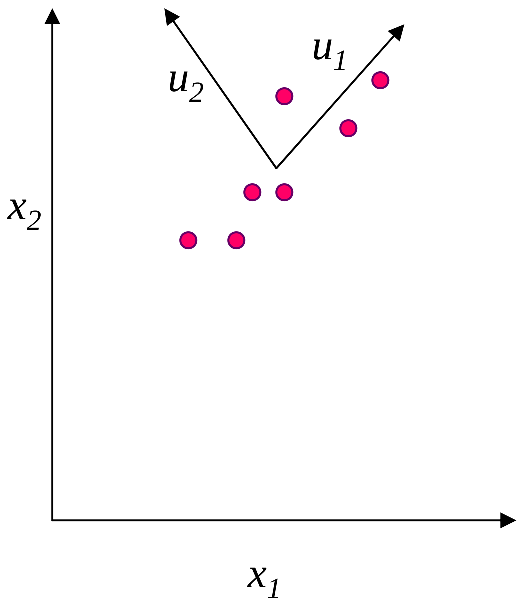
PCA: given M<d. Find $\langle \mathbf{u}_1 \ldots \mathbf{u}_M \rangle$

that minimizes $E_M \equiv \sum_{n=1}^{N} ||\mathbf{x}^n - \hat{\mathbf{x}}^n||^2$

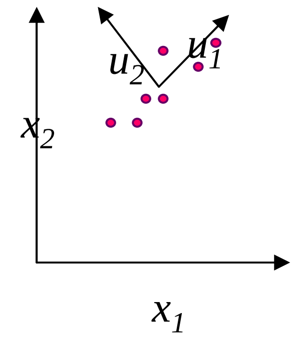
where $\hat{\mathbf{x}}^n = \bar{\mathbf{x}} + \sum_{i=1}^{M} z_i^n \mathbf{u}_i$

Mean

$$\bar{\mathbf{x}} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}^n$$



PCA



PCA: given M<d. Find $\langle \mathbf{u}_1 \ldots \mathbf{u}_M \rangle$

that minimizes $E_M \equiv \sum_{n=1}^{N} ||\mathbf{x}^n - \hat{\mathbf{x}}^n||^2$

where $\hat{\mathbf{x}}^n = \bar{\mathbf{x}} + \sum_{i=1}^{M} z_i^n \mathbf{u}_i$

Note we get zero error if M=d.

Therefore, $E_M = \sum_{i=M+1}^{d} \sum_{n=1}^{N} [\mathbf{u}_i^T(\mathbf{x}^n - \bar{\mathbf{x}})]^2$

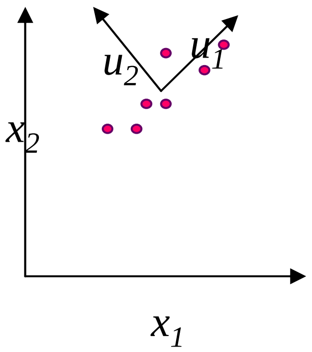
$$= \sum_{i=M+1}^{d} \mathbf{u}_i^T \Sigma \ \mathbf{u}_i$$

This minimized when $\boldsymbol{u}_i$ is eigenvector of Σ, i.e., when:

$$\Sigma \mathbf{u}_i = \lambda_i \mathbf{u}_i$$

Covariance matrix: $\Sigma = \sum_n (\mathbf{x}^n - \bar{\mathbf{x}})(\mathbf{x}^n - \bar{\mathbf{x}})^T$

PCA



Minimize $E_M = \sum_{i=M+1}^{d} \mathbf{u}_i^T \mathbf{\Sigma} \ \mathbf{u}_i$

$\rightarrow \quad \mathbf{\Sigma u}_i = \lambda_i \mathbf{u}_i$

Eigenvector of Σ

Eigenvalue

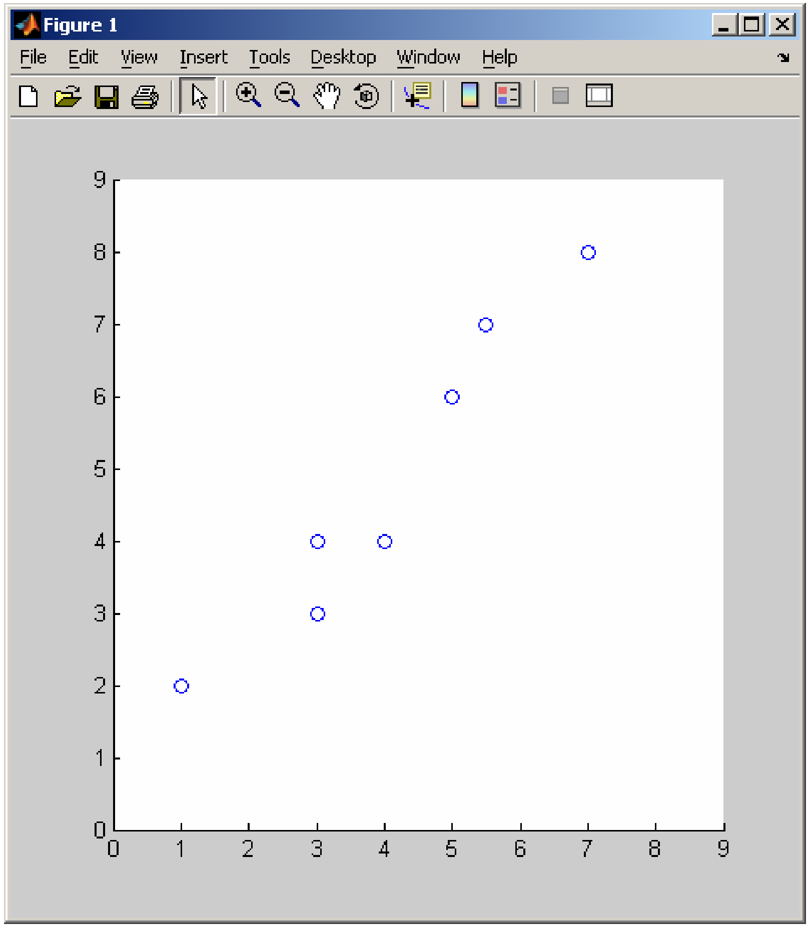
$\rightarrow \quad E_M = \sum_{i=M+1}^{d} \lambda_i$

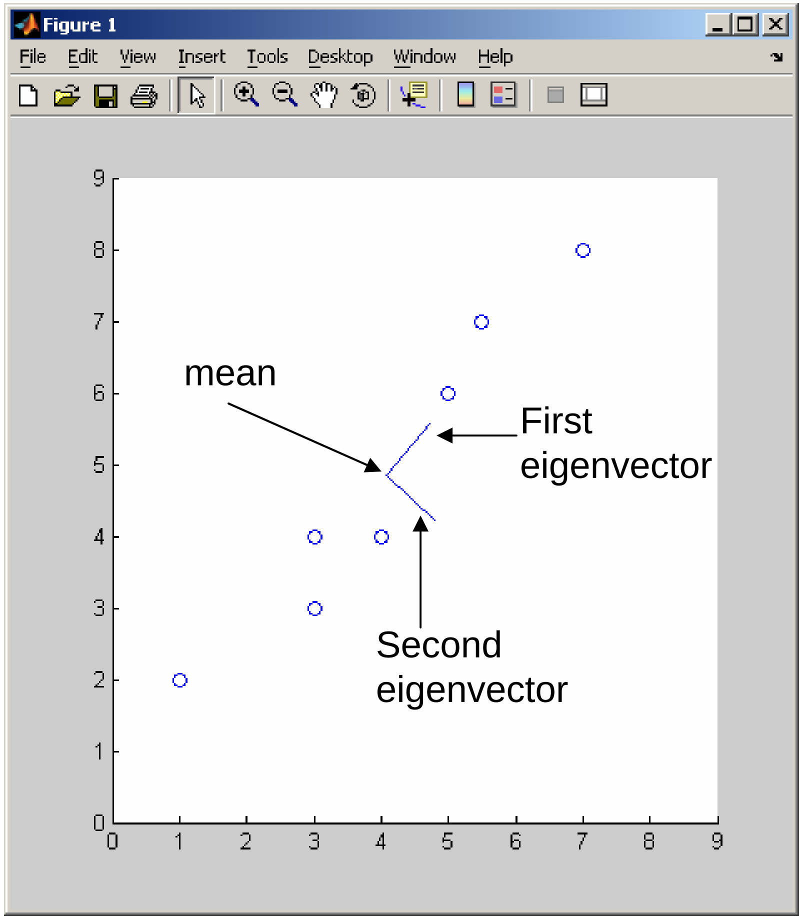
PCA algorithm 1:

1. X ← Create N x d data matrix, with one row vector x^n per data point
2. X ← subtract mean $\bar{x}$ from each row vector x^n in X
3. Σ ← covariance matrix of X
4. Find eigenvectors and eigenvalues of Σ
5. PC's ← the M eigenvectors with largest eigenvalues

PCA Example

$$\hat{\mathbf{x}}^n = \bar{\mathbf{x}} + \sum_{i=1}^{M} z_i^n \mathbf{u}_i$$

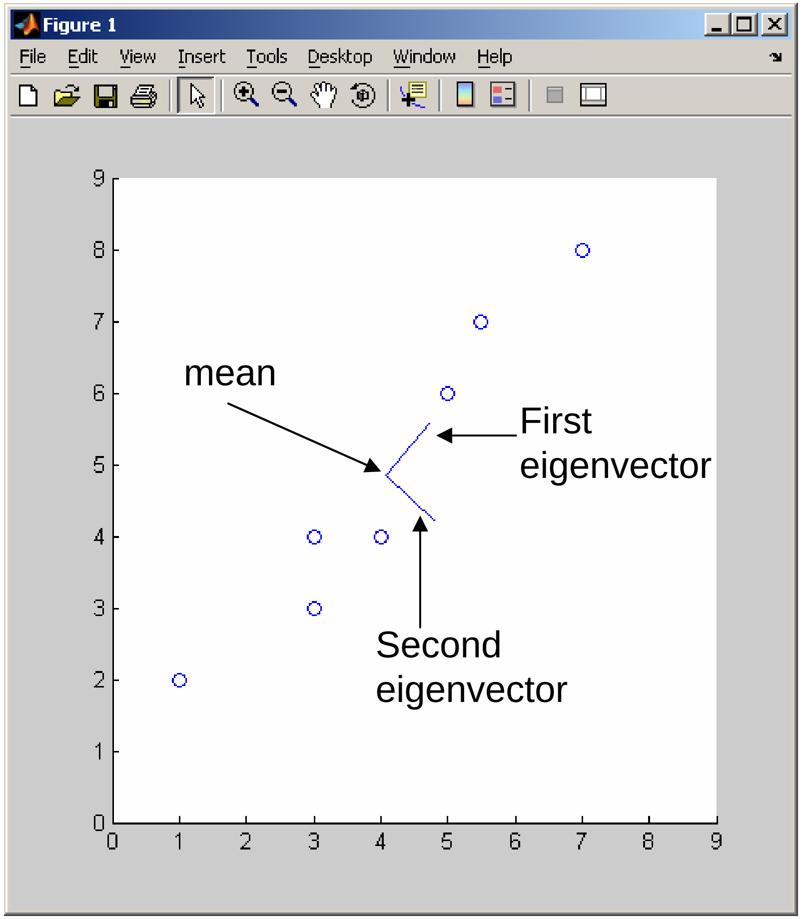


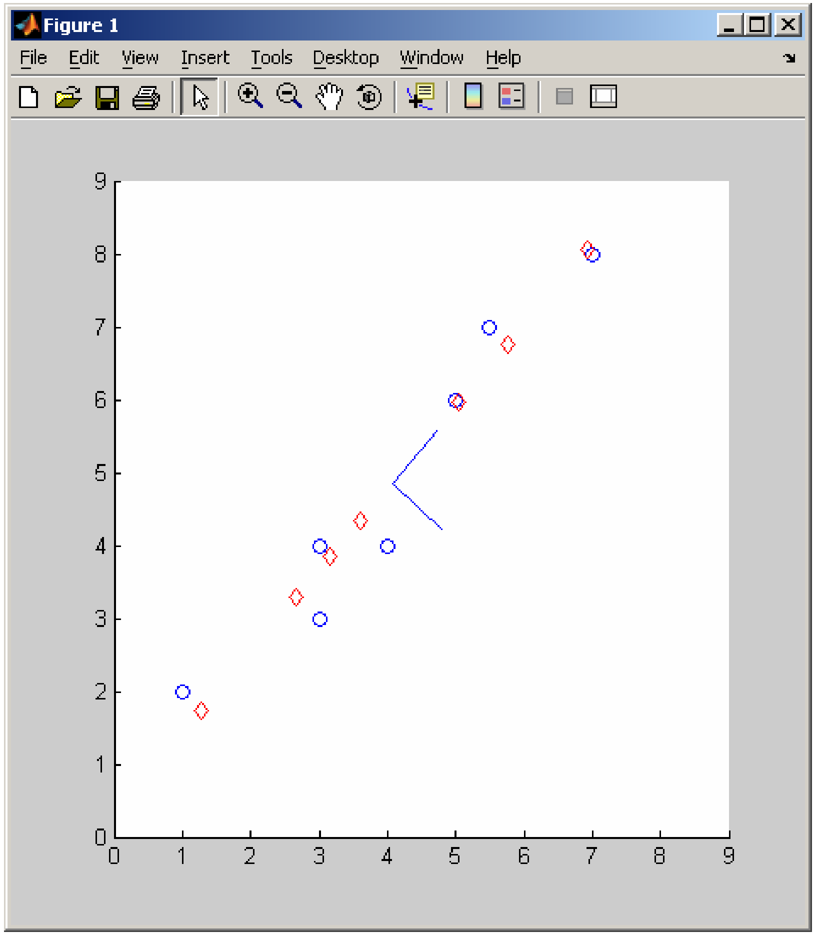


PCA Example

$$\hat{\mathbf{x}}^n = \bar{\mathbf{x}} + \sum_{i=1}^{M} z_i^n \mathbf{u}_i$$

Reconstructed data using only first eigenvector (M=1)





Very Nice When Initial Dimension Not Too Big

What if very large dimensional data?

- e.g., Images (d $\geq$ 10^4)

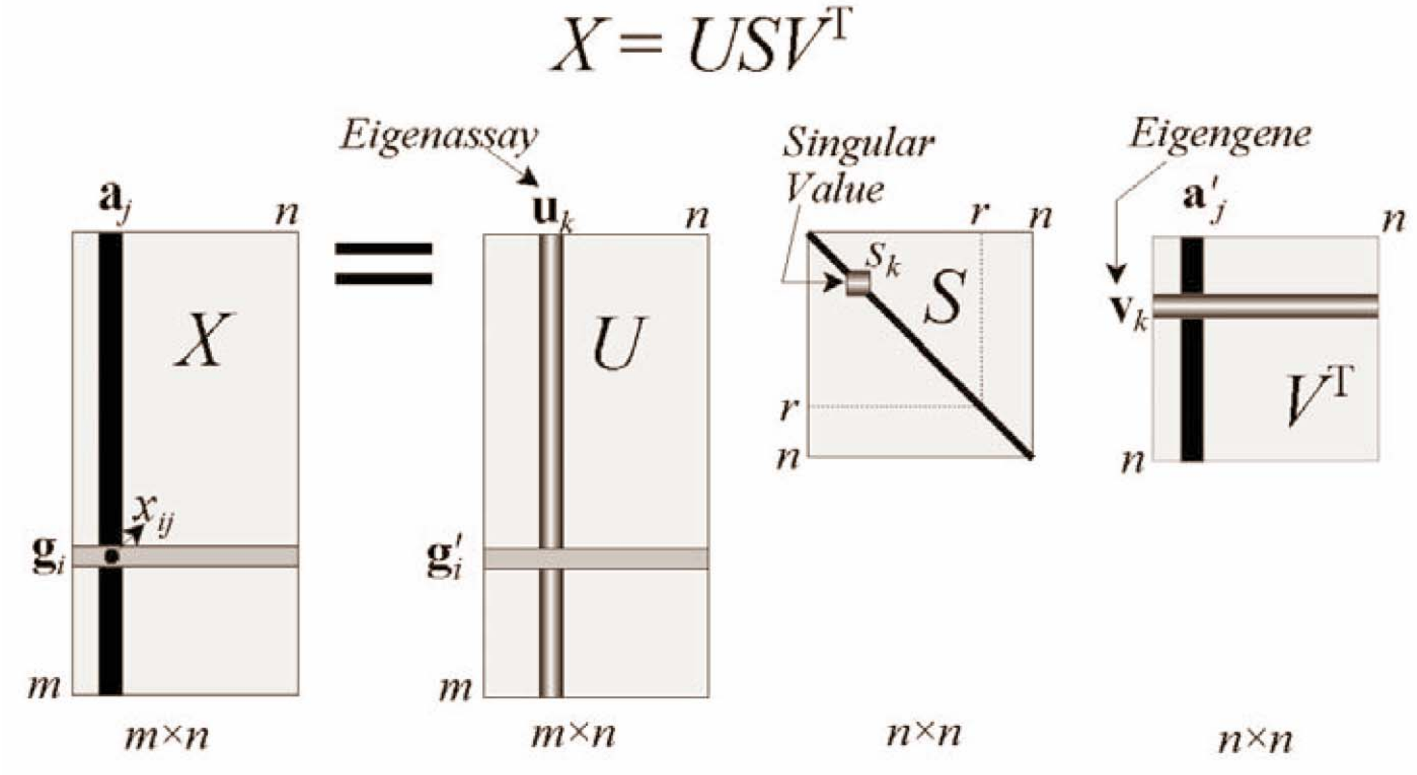
Problem:

- Covariance matrix Σ is size (d x d)
- $d=10^4 \rightarrow |\Sigma| = 10^8$

Singular Value Decomposition (SVD) to the rescue!

- pretty efficient algs available, including Matlab SVD
- some implementations find just top N eigenvectors

SVD



Data X, one row per data point

US gives coordinates of rows of X in the space of principle components

S is diagonal, $S_k > S_{k+1}$, S_k^2 is kth largest eigenvalue

Rows of V^T are unit length eigenvectors of X^TX.

If cols of X have zero mean, then $X^TX = c\,\Sigma$ and eigenvects are the Principle Components

[from Wall et al., 2003]

Singular Value Decomposition

To generate principle components:

- Subtract mean $\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}^n$ from each data point, to create zero-centered data
- Create matrix X with one row vector per (zero centered) data point
- Solve SVD: $X = USV^T$
- Output Principle components: columns of V (= rows of V^T)
  - Eigenvectors in V are sorted from largest to smallest eigenvalues
  - S is diagonal, with s_k^2 giving eigenvalue for kth eigenvector

Singular Value Decomposition

To project a point (column vector x) into PC coordinates:

$V^T x$

If x_i is i[th] row of data matrix X, then

- (i[th] row of US) $= V^T x_i^T$
- $(US)^T = V^T X^T$

To project a column vector x to M dim Principle Components subspace, take just the first M coordinates of $V^T x$

Independent Components Analysis

- PCA seeks directions <$Y_1 \ldots Y_M$> in feature space X that minimize reconstruction error

- ICA seeks directions <$Y_1 \ldots Y_M$> that are most *statistically independent*. I.e., that minimize $I(Y)$, the mutual information between the Y_j :

$$I(Y) = \left[\sum_{j=1}^{J} H(Y_j) \right] - H(Y)$$

Which maximizes their departure from Gaussianity!

Independent Components Analysis

- ICA seeks to minimize *I(Y)*, the mutual information between the Y_j :

$$I(Y) = \left[\sum_{j=1}^{J} H(Y_j) \right] - H(Y)$$

$$\begin{vmatrix} y_1(t) \\ \vdots \\ y_m(t) \end{vmatrix} = \mathbf{W} \begin{vmatrix} x_1(t) \\ \vdots \\ x_n(t) \end{vmatrix}$$

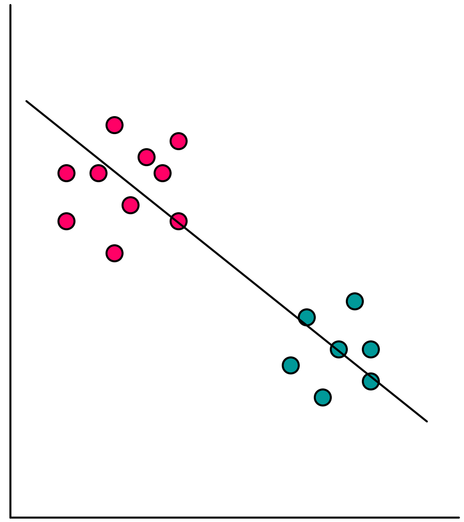
- Example: Blind source separation
  - Original features $x_i(t)$ are microphones at a cocktail party
  - Each receives sounds from multiple people speaking
  - ICA outputs directions that correspond to individual speakers $y_k(t)$

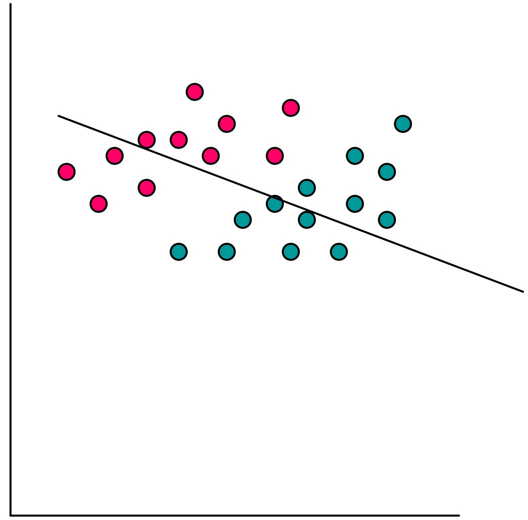
Supervised Dimensionality Reduction

1. Fisher Linear Discriminant

- A method for projecting data into lower dimension to hopefully improve classification

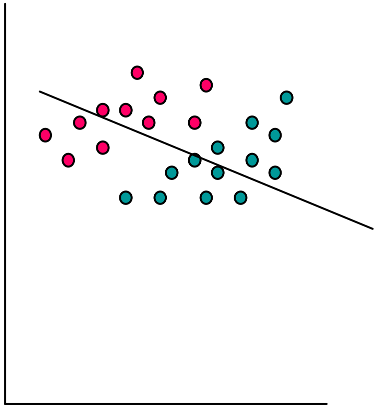
- We'll consider 2-class case





Project data onto vector that connects class means?

Fisher Linear Discriminant



Project data onto one dimension, to help classification

$$y = \mathbf{w}^T \mathbf{x}$$

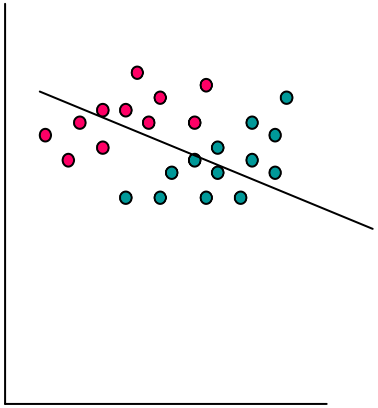
Define class means: $\mathbf{m}_i = \frac{1}{N_i} \sum_{n \in C_i} \mathbf{x}^n$

Could choose w according to: $\arg\max_{\mathbf{w}} \mathbf{w}^T (\mathbf{m_2} - \mathbf{m_1})$

Instead, Fisher Linear Discriminant chooses: $\arg\max_{\mathbf{w}} \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}$

$$m_i \equiv \mathbf{w}^T \mathbf{m}_i \qquad s_i^2 \equiv \sum_{n \in C_i} (y^n - m_i)^2$$

Fisher Linear Discriminant



Project data onto one dimension, to help classification

$$y = \mathbf{w}^T \mathbf{x}$$

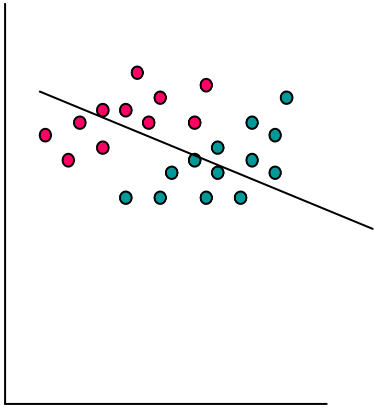
Fisher Linear Discriminant : $\arg\max_{\mathbf{w}} \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}$

is solved by : $\mathbf{w} \propto \mathbf{S_W}^{-1}(\mathbf{m_2} - \mathbf{m_1})$

Where S_W is sum of within-class covariances:

$$\mathbf{S_W} \equiv \sum_{n \in C_1} (\mathbf{x}^n - \mathbf{m_1})(\mathbf{x}^n - \mathbf{m_1})^T + \sum_{n \in C_2} (\mathbf{x}^n - \mathbf{m_2})(\mathbf{x}^n - \mathbf{m_2})^T$$

Fisher Linear Discriminant



Fisher Linear Discriminant : $\arg\max_{\mathbf{w}} \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}$

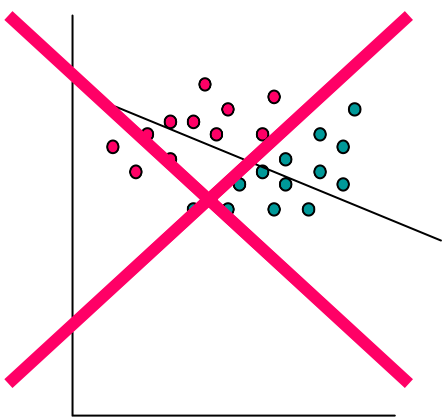
Is equivalent to minimizing sum of squared error if we assume target values are not +1 and -1, but instead N/N_1 and $-N/N_2$

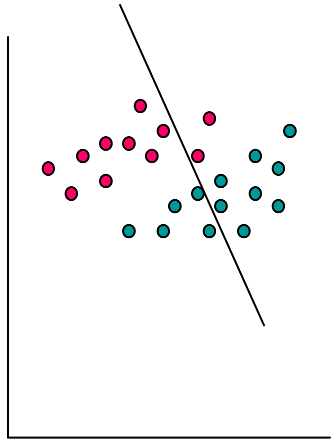
Where N is total number of examples, N_i is number in class i

Also generalized to K classes (and projects data to K-1 dimensions)

Summary: Fisher Linear Discriminant

- Choose n-1 dimension projection for n-class classification problem
- Use within-class covariances to determine the projection
- Minimizes a different sum of squared error function

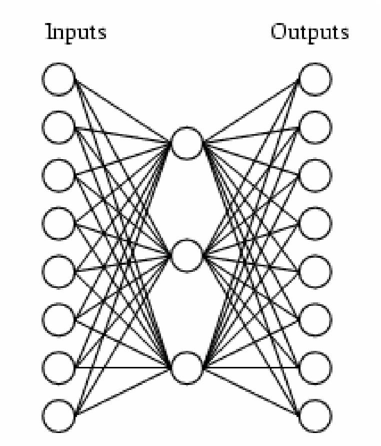




2. Hidden Layers in Neural Networks

When # hidden units < # inputs, hidden layer also performs dimensionality reduction.

Each synthesized dimension (each hidden unit) is logistic function of inputs



$$h_k(\mathbf{x}) = \frac{1}{1 + exp(w_0 + \sum_{i=1}^{N} w_i x_i)}$$

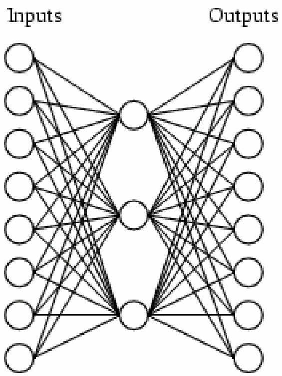
Hidden units defined by gradient descent to (locally) minimize squared output classification/regression error

$$E = \sum_{n=1}^{N} \sum_{k} (\hat{y_k}(x^n) - y_k(x^n))^2$$

Also allow networks with multiple hidden layers

→ highly nonlinear components (in contrast with linear subspace of Fisher LD, PCA)

Learning Hidden Layer Representations



Training neural network to minimize reconstruction error

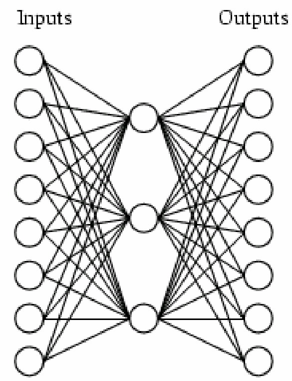
A target function:

Input		Output
10000000	→	10000000
01000000	→	01000000
00100000	→	00100000
00010000	→	00010000
00001000	→	00001000
00000100	→	00000100
00000010	→	00000010
00000001	→	00000001

Can this be learned??

Learning Hidden Layer Representations

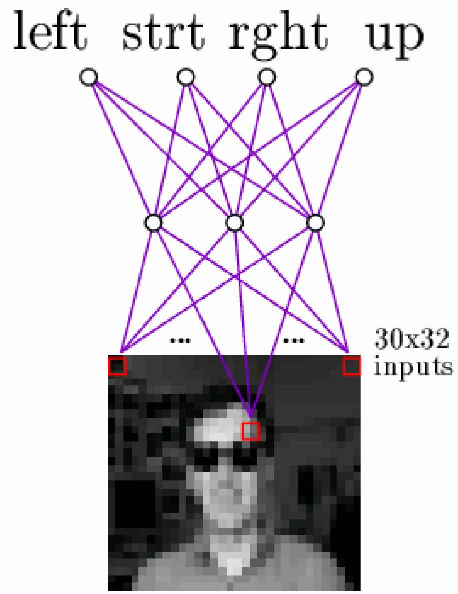
A network:



Learned hidden layer representation:

Input		Hidden Values				Output
10000000	→	.89	.04	.08	→	10000000
01000000	→	.01	.11	.88	→	01000000
00100000	→	.01	.97	.27	→	00100000
00010000	→	.99	.97	.71	→	00010000
00001000	→	.03	.05	.02	→	00001000
00000100	→	.22	.99	.99	→	00000100
00000010	→	.80	.01	.98	→	00000010
00000001	→	.60	.94	.01	→	00000001

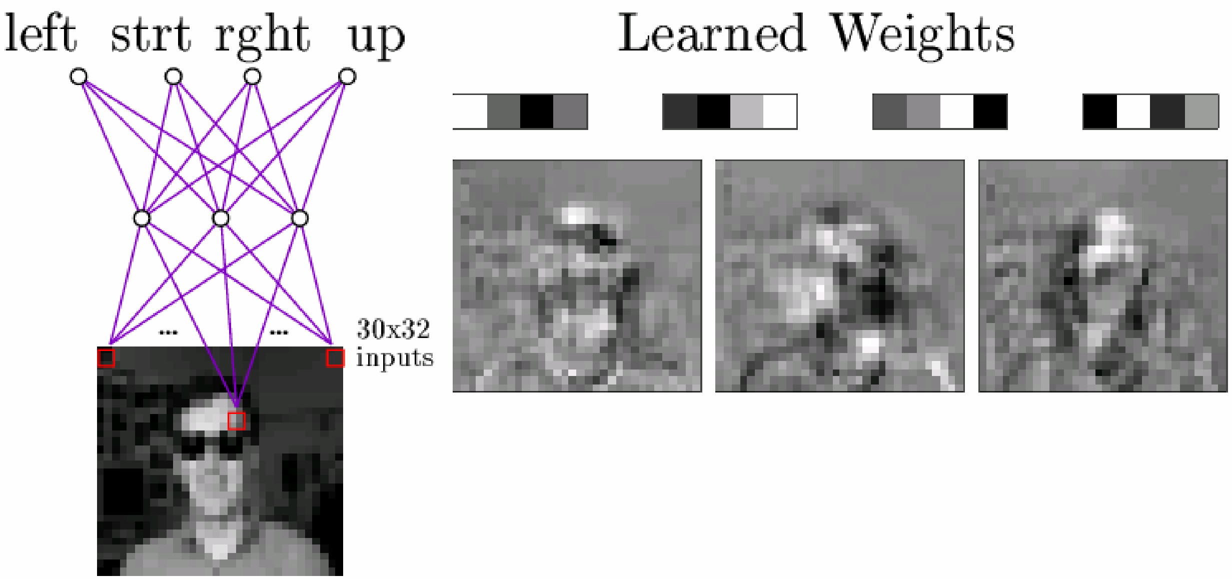
Neural Nets for Face Recognition

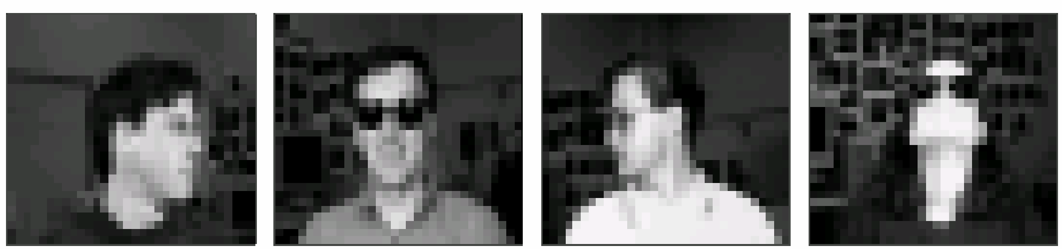


Typical input images

90% accurate learning head pose, and recognizing 1-of-20 faces

Learned Hidden Unit Weights



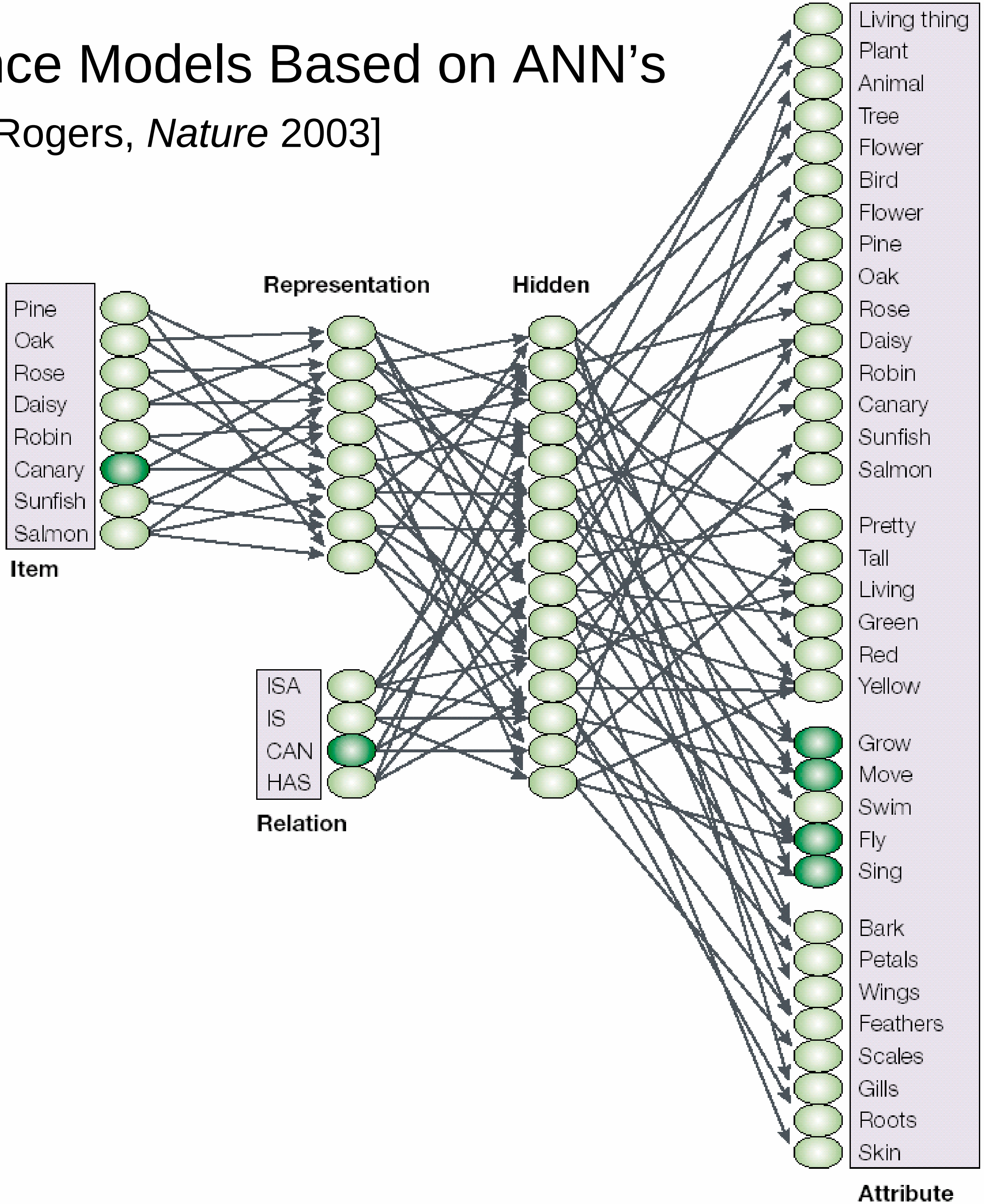


Typical input images

http://www.cs.cmu.edu/~tom/faces.html

Cognitive Neuroscience Models Based on ANN's

[McClelland & Rogers, *Nature* 2003]



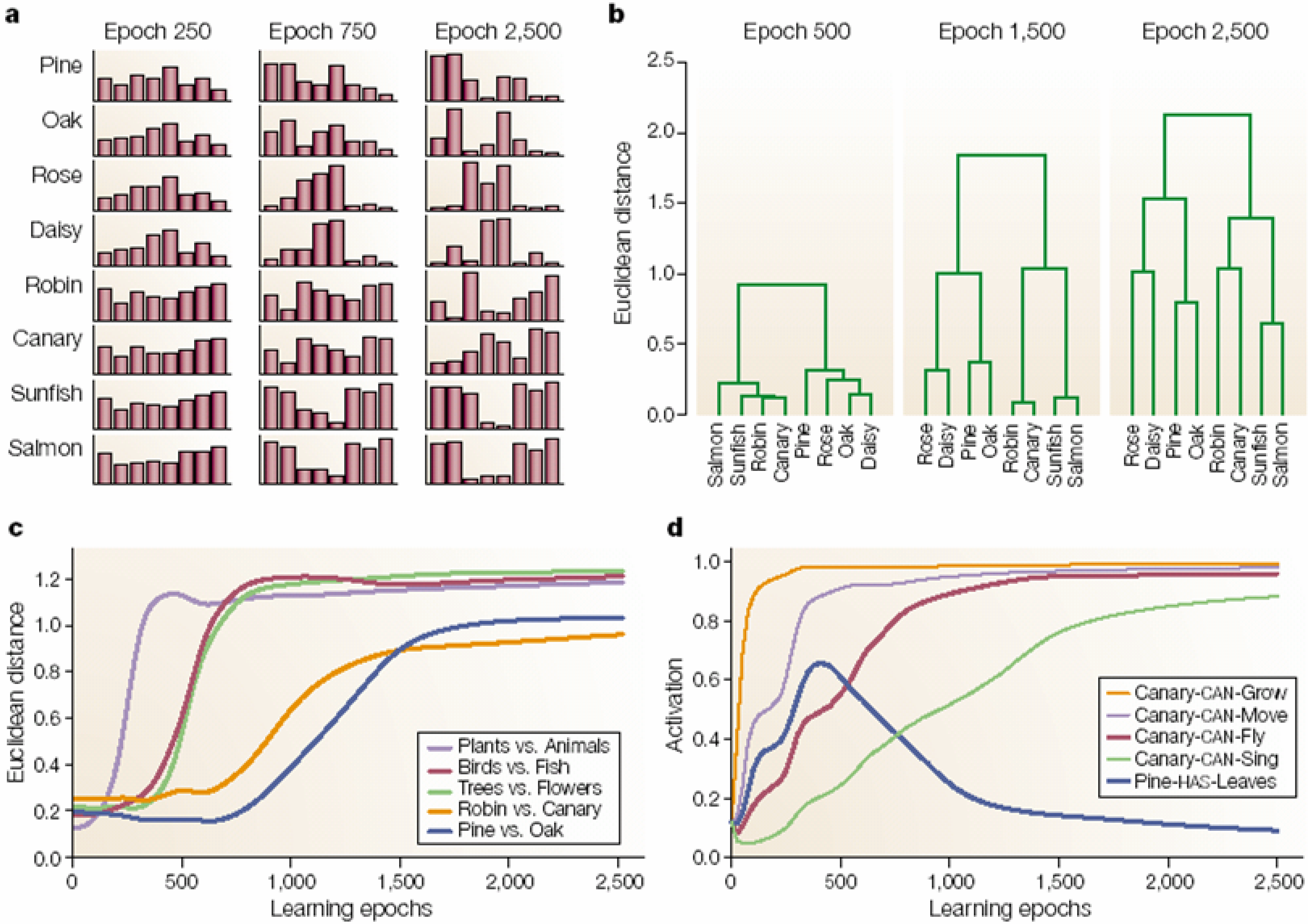


Figure 4 | **The process of differentiation of conceptual representations.** The representations are those seen in the feedforward network model shown in FIG. 3. **a** | Acquired patterns of activation that represent the eight objects in the training set at three points in the learning process (epochs 250, 750 and 2,500). Early in learning, the patterns are undifferentiated; the first difference to appear is between plants and animals. Later, the patterns show clear differentiation at both the superordinate (plant–animal) and intermediate (bird–fish/tree–flower) levels. Finally, the individual concepts are differentiated, but the overall hierarchical organization of the similarity structure remains. **b** | A standard hierarchical clustering analysis program has been used to visualize the similarity structure in the

What you should know

- Feature selection
  - Single feature scoring criteria
  - Search strategies
    - Common approaches: Greedy addition of features, or greedy deletion

- Unsupervised dimension reduction using all features
  - Principle Components Analysis
    - Minimize reconstruction error
  - Singular Value Decomposition
    - Efficient PCA
  - Independent components analysis

- Supervised dimension reduction
  - Fisher Linear Discriminant
    - Project to n-1 dimensions to discriminate n classes
  - Hidden layers of Neural Networks
    - Most flexible, local minima issues

Further Readings

- “Singular value decomposition and principal component analysis,” Wall, M.E, Rechtsteiner, A., and L. Rocha, in *A Practical Approach to Microarray Data Analysis* (D.P. Berrar, W. Dubitzky, M. Granzow, eds.) Kluwer, Norwell, MA, 2003. pp. 91-109.

  LANL LA-UR-02-4001