Today: Learning representations II
- Artificial neural networks
- PCA
- ICA
- CCA

Readings:
- Bishop Ch. 12 through 12.1
- “A Tutorial on PCA,” J. Schlens
- Wall et al., 2003

Neural Nets for Face Recognition

90% accurate learning head pose, and recognizing 1-of-20 faces
Semantic Memory Model Based on ANN's
[McClelland & Rogers, Nature 2003]

Train with assertions, e.g., Can(Canary,Fly)
Humans act as though they have a hierarchical memory organization

1. Victims of Semantic Dementia progressively lose knowledge of objects. But they lose specific details first, general properties later, suggesting hierarchical memory organization.

    | Thing | NonLiving | Living | Plant | Animal |
    |-------|-----------|--------|-------|--------|
    | Fish  | Bird      | Canary |
    |       |           |        |

2. Children appear to learn general categories and properties first, following the same hierarchy, top down.

Question: What learning mechanism could produce this emergent hierarchy?

* some debate remains on this.

---

Memory deterioration follows semantic hierarchy

(a) Picture naming responses for JL

<table>
<thead>
<tr>
<th>Item</th>
<th>Sept. '91</th>
<th>March '92</th>
<th>March '93</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bird</td>
<td>+</td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>Chicken</td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Duck</td>
<td>+</td>
<td>Bird</td>
<td>Dog</td>
</tr>
<tr>
<td>Swan</td>
<td>+</td>
<td>Bird</td>
<td>Animal</td>
</tr>
<tr>
<td>Eagle</td>
<td>Duck</td>
<td>Bird</td>
<td>Horse</td>
</tr>
<tr>
<td>Ostrich</td>
<td>Swan</td>
<td>Bird</td>
<td>Animal</td>
</tr>
<tr>
<td>Peacock</td>
<td>Duck</td>
<td>Bird</td>
<td>Vehicle</td>
</tr>
<tr>
<td>Penguin</td>
<td>Duck</td>
<td>Bird</td>
<td>Part of animal</td>
</tr>
<tr>
<td>Rooster</td>
<td>Chicken</td>
<td>Chicken</td>
<td>Dog</td>
</tr>
</tbody>
</table>

(b) [McClelland & Rogers, *Nature* 2003]

(c) IF’s delayed copy of a camel

(d) DC’s delayed copy of a swan
ANN Also Models Progressive Deterioration

[McClelland & Rogers, Nature 2003]
Training Networks on Time Series

• Suppose we want to predict next state of world
  – and it depends on history of unknown length
  – e.g., robot with forward-facing sensors trying to predict next sensor reading as it moves and turns

• Idea: use hidden layer in network to capture state history

(a) Feedforward network  
(b) Recurrent network
Training Networks on Time Series

How can we train recurrent net??

Summary: Neural Networks

- Represent highly non-linear decision surfaces
- Learn $f: X \rightarrow Y$, where $Y$ is vector (e.g., image)
- Hidden layer represents re-representation of input
  - to optimize prediction accuracy (minimize sum sq error)
- Role in modeling human cognition
- Local minimum problems solving for MLE/MAP parameters using gradient descent
Learning Lower Dimensional Representations

- Supervised learning of lower dimension representation
  - Hidden layers in Neural Networks
  - Fisher linear discriminant

- Unsupervised learning of lower dimension representation
  - Principle Components Analysis (PCA)
  - Independent components analysis (ICA)
  - Canonical correlation analysis (CCA)

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
Principle Components Analysis

- Like auto-encoding neural networks, learn re-representation of input data that can best reconstruct it.
  - Learned encoding is linear function of inputs (not logistic).
  - No local minimum problems when training!
  - Given d-dimensional data \( X \), learns d-dimensional representation, where
    - the dimensions are orthogonal
    - top \( k \) dimensions are the \( k \)-dimensional linear re-representation that minimizes reconstruction error (sum of squared errors).

---

PCA Example

\[
\text{face}_i = \sum_k c_{ik} \text{eigenface}_k
\]

Thanks to Christopher DeCoro

Reconstructing a face from the first N components (eigenfaces)

Adding 1 additional PCA component at each step

Adding 8 additional PCA components at each step

In this next image, we show a similar picture, but with each additional face representing an additional 8 principle components. You can see that it takes a rather large number of images before the picture looks totally correct.

**Learned Hidden Unit Weights**

Typical input images

http://www.cs.cmu.edu/~tom/faces.html
PCA: Find Projections to Minimize Reconstruction Error

Assume data is set of d-dimensional vectors, where nth vector is
\[ x^n = (x^n_1 \ldots x^n_d) \]

We can represent these in terms of any d orthogonal vectors \( u_1 \ldots u_d \)
\[ x^n = \sum_{i=1}^{d} z^n_i u_i; \quad u_i^T u_j = \delta_{ij} \]

**PCA:** given \( M < d \). Find \( \langle u_1 \ldots u_M \rangle \)
that minimizes \( E_M = \sum_{n=1}^{N} ||x^n - \bar{x}^n||^2 \)
where \( \bar{x}^n = \bar{x} + \sum_{i=1}^{M} z^n_i u_i \)

\[ \bar{x} = \frac{1}{N} \sum_{n=1}^{N} x^n \]

Note we get zero error if \( M = d \), so all error is due to missing components.

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

Covariance matrix:
\[ \Sigma = \sum_{n=1}^{N} (x^n - \bar{x})(x^n - \bar{x})^T \]
\[ \Sigma_{ij} = \sum_{n=1}^{N} (x^n_i - \bar{x}_i)(x^n_j - \bar{x}_j) \]

This minimized when \( u_i \) is eigenvector of \( \Sigma \), the covariance matrix of \( X \).

i.e., minimized when:
\[ \Sigma u_i = \lambda_i u_i \]
PCA

Minimize \( E_M = \sum_{i=M+1}^{d} u_i^T \Sigma u_i \)

\( \rightarrow \Sigma u_i = \lambda_i u_i \quad \text{Eigenvector of} \ \Sigma \)

\( \text{Eigenvalue (scalar)} \)

\( \rightarrow E_M = \sum_{i=M+1}^{d} \lambda_i \)

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

PCA Example

\( \bar{x}^n = \bar{x} + \sum_{i=1}^{M} z_i^n u_i \)
PCA Example

\[ \bar{x}^n = \bar{x} + \sum_{i=1}^{M} z_i^n u_i \]

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, \(10^4\))

Problem:
- Covariance matrix \(\Sigma\) 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**

\[
X = USV^T
\]

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 \(X^TX = \Sigma\) and eigenvects are the Principle Components

**Singular Value Decomposition**

To generate principle components:

- Subtract mean \(\bar{x} = \frac{1}{N} \sum_{n=1}^{N} x^n\) 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 \(VT\))
  - 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 (ICA)

• PCA seeks orthogonal 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 \( Y_i \):

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