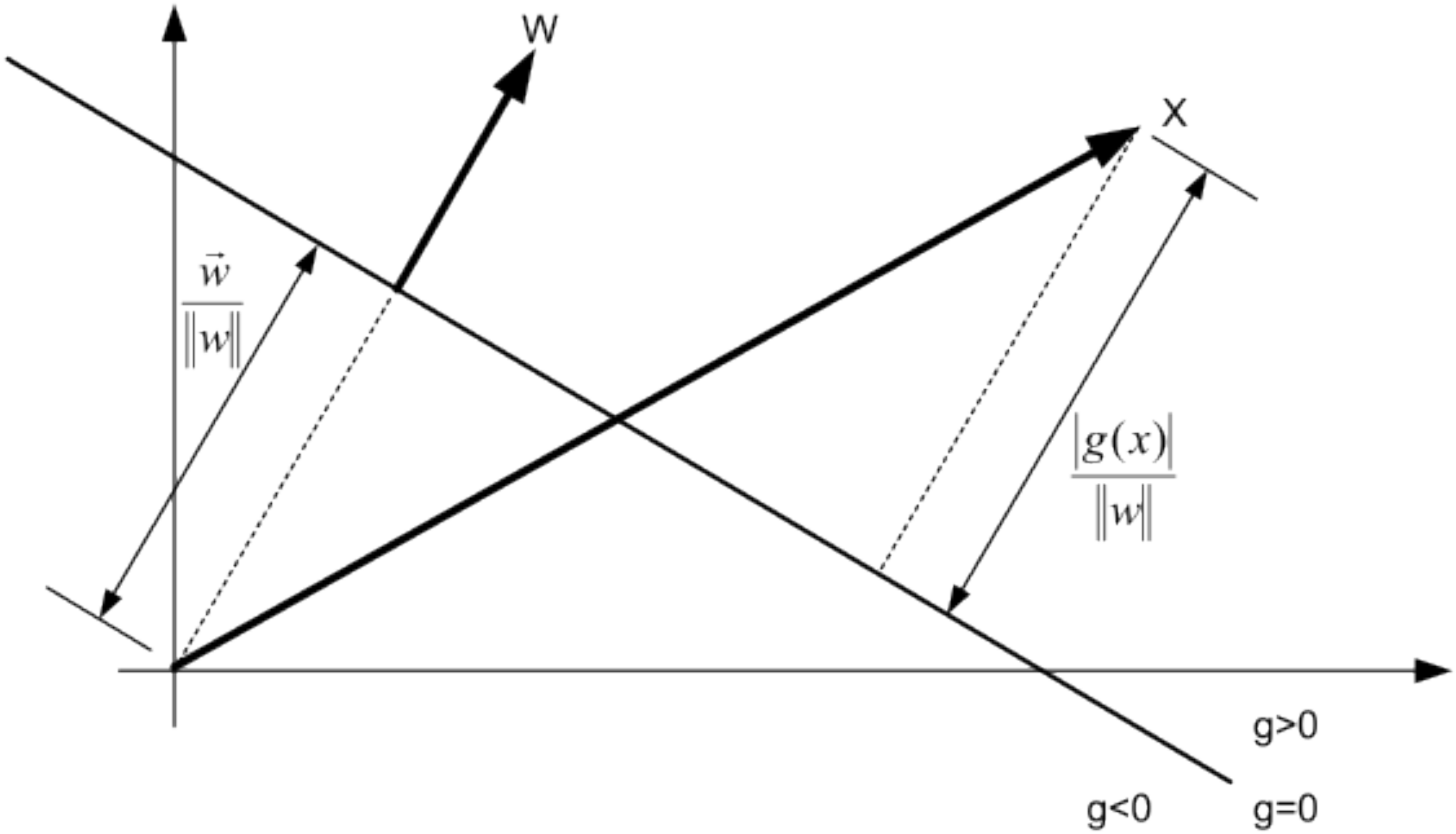
Neural Nets, Background
Back-propagation

Neuronale Netze u. Anwendungen, 19. 11. 2011

Linear Discriminant Function I



Linear Discriminant Functions II

Hyperplane H: $g(\vec{x}) = \sum_{i=1}^{n} w_1 x_1 + w_0 = \vec{w}^T \vec{x} + w_0 = 0$

$$\Rightarrow \quad \vec{x} = q \frac{\vec{w}}{\|\vec{w}\|} + r \frac{\vec{w}}{\|\vec{w}\|} + \vec{x}_p$$

Vector $q \frac{\vec{w}}{\|\vec{w}\|}$ equals: $g\left(q \frac{\vec{w}}{\|\vec{w}\|} \right) = 0 = q\|\vec{w}\| + w_0 \Rightarrow \boxed{q = -\frac{w_0}{\|\vec{w}\|}}$

And with $g(\vec{x}) = \vec{w}^T q \frac{\vec{w}}{\|\vec{w}\|} + \vec{w}^T r \frac{\vec{w}}{\|\vec{w}\|} + \vec{w}\vec{x}_p + w_0 = -w_0 + r\|\vec{w}\| + w_0$

We get $\boxed{r = \frac{g(\vec{x})}{\|\vec{w}\|}}$

Using Neural Nets

- Classification
- Prediction
- Function Approximation
- Continuous Mapping
- Pattern Completion
- Coding

Design Criteria

- Recognition Error Rate
- Training Time
- Recognition Time
- Memory Requirements
- Training Complexity
- Ease of Implementation
- Ease of Adaptation

Network Specification

- What parameters are typically chosen by the network designer:
  - Net Topology
  - Node Characteristics
  - Learning Rule
  - Objective Function
  - (Initial) Weights
  - Learning Parameters

Neural Models

- Back-Propagation
- Boltzman Machines
- Decision Tree Classifiers
- Feature Map Classifiers
- Learning Vector Quantizer (LVQ, LVQ2)
- High Order Networks
- Radial Basis Functions
- Modified Nearest Neighbor
- ART
- etc.

Applications

- Space Robot*
- Autonomous Navigation*
- Speech Recognition and Understanding*
- Natural Language Processing*
- Music*
- Gesture Recognition
- Lip Reading
- Face Recognition
- Household Robots
- Signal Processing
- Banking, Bond Rating,...
- Sonar
- etc....

Advanced Neural Models

- Time-Delay Neural Networks (Waibel)
- Recurrent Nets (Elman, Jordan)
- Higher Order Nets
- Modular System Construction
- Adaptive Architectures

Neural Nets - Design Problems

- Local Minima
- Speed of Learning
- Architecture must be selected
- Choice of Feature Representation
- Scaling
- Systems, Modularity
- Treatment of Temporal Features and Sequences

Neural Nets - Modeling

- Neural Nets are
  - Non-Linear Classifiers
  - Approximate Posterior Probabilities
  - Non-Parametric Training
- The Problem with Time
  - How to Consider Context
  - How to Operate Shift-Invariantly
  - How to Process Pattern Sequences

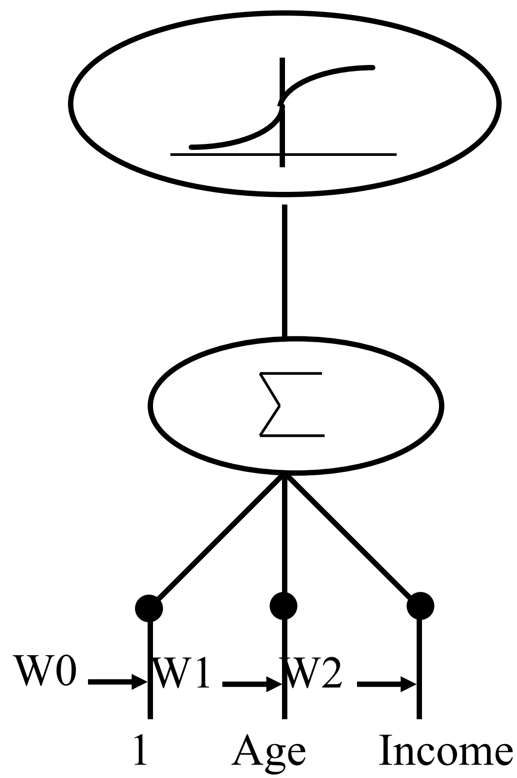
Time Varying Patterns

- Detect B in Context A
  - AAABAAA ---> 1
  - AAABCAA ---> 0
- Detect B Independent of Point in Time
  - AAAABAAAAAAA ---> 1
  - AAAAAAAABAAA ---> 1
  - AAAAAAAAAAAA ----> 0
- Detect Sequence ABC
  - AAAAAABBCCCC ---> 1
  - ABBBBBBCCC ---> 1
  - CCAAABB ---> 0

Time-Delay Neural Networks

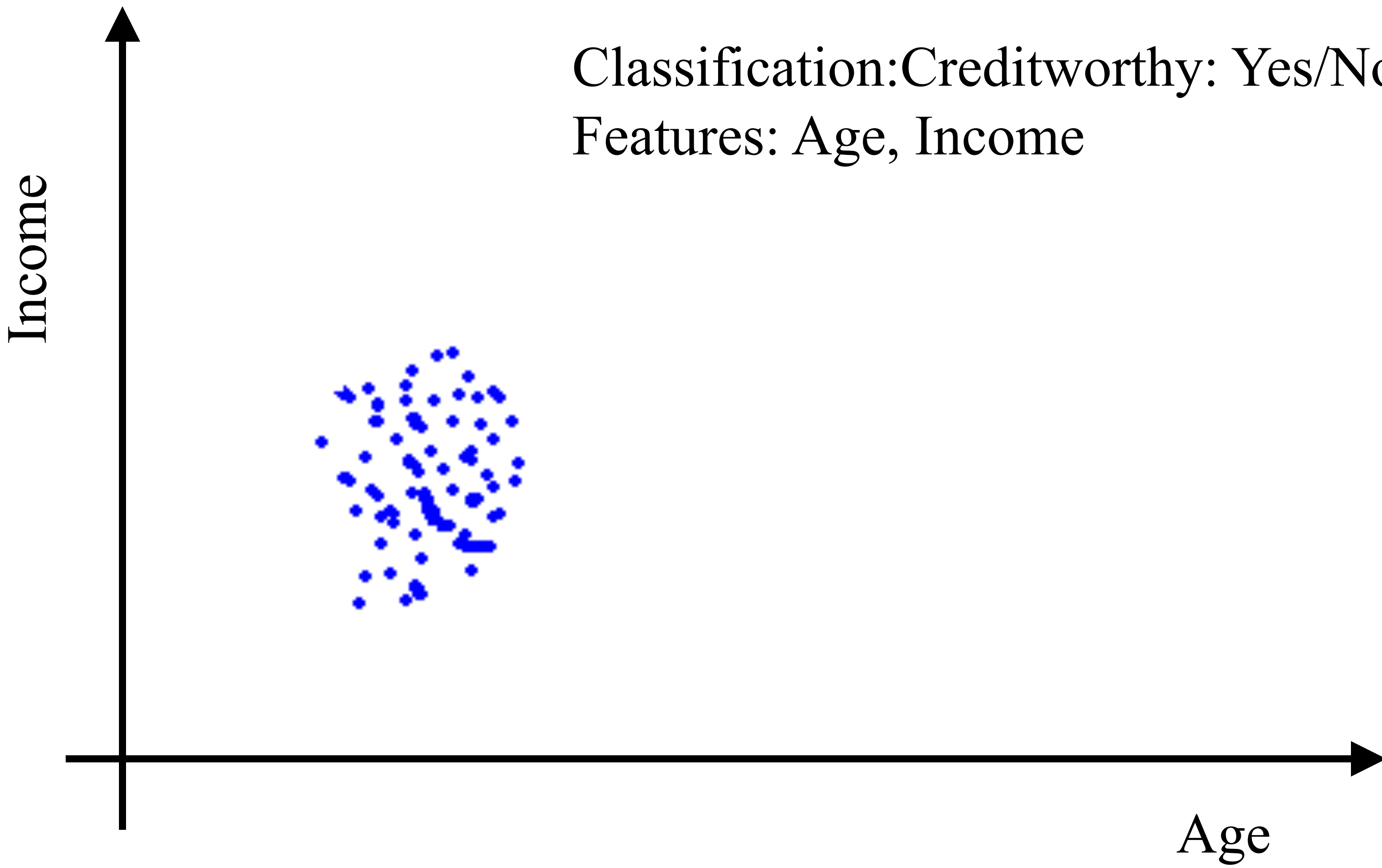
- Multilayer Neural Network - Nonlinear Classifier
- Time-Delays on Connections
  - At Input Layer
  - At all Layers
- Shift-Invariant Learning
  - All Units Learn to Detect Patterns Independent of Location in Time
  - No Presegmentation or Prealignment Necessary
  - Approach: Weight Sharing

The Perceptron



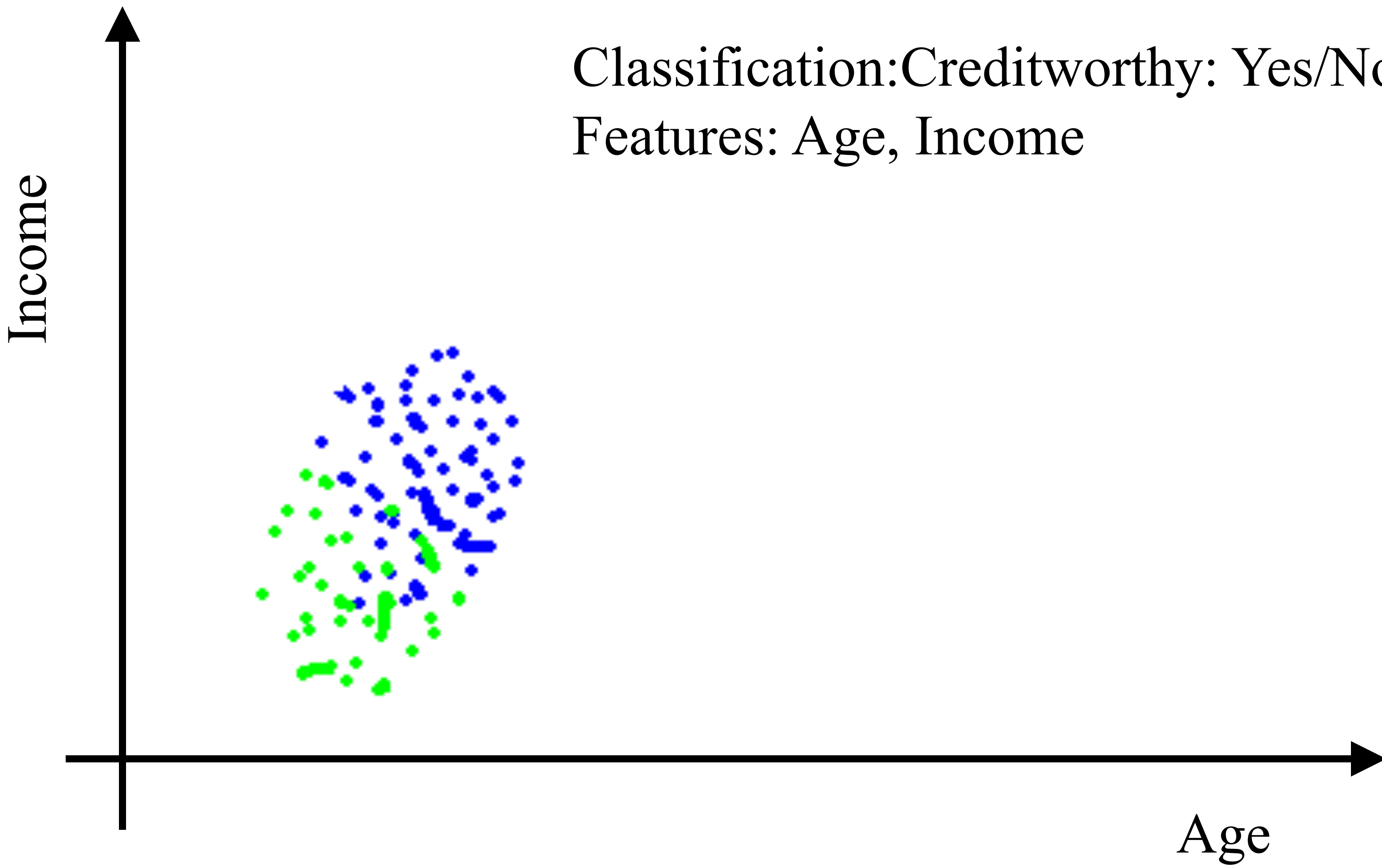
Pattern recognition

Classification:Creditworthy: Yes/No
Features: Age, Income



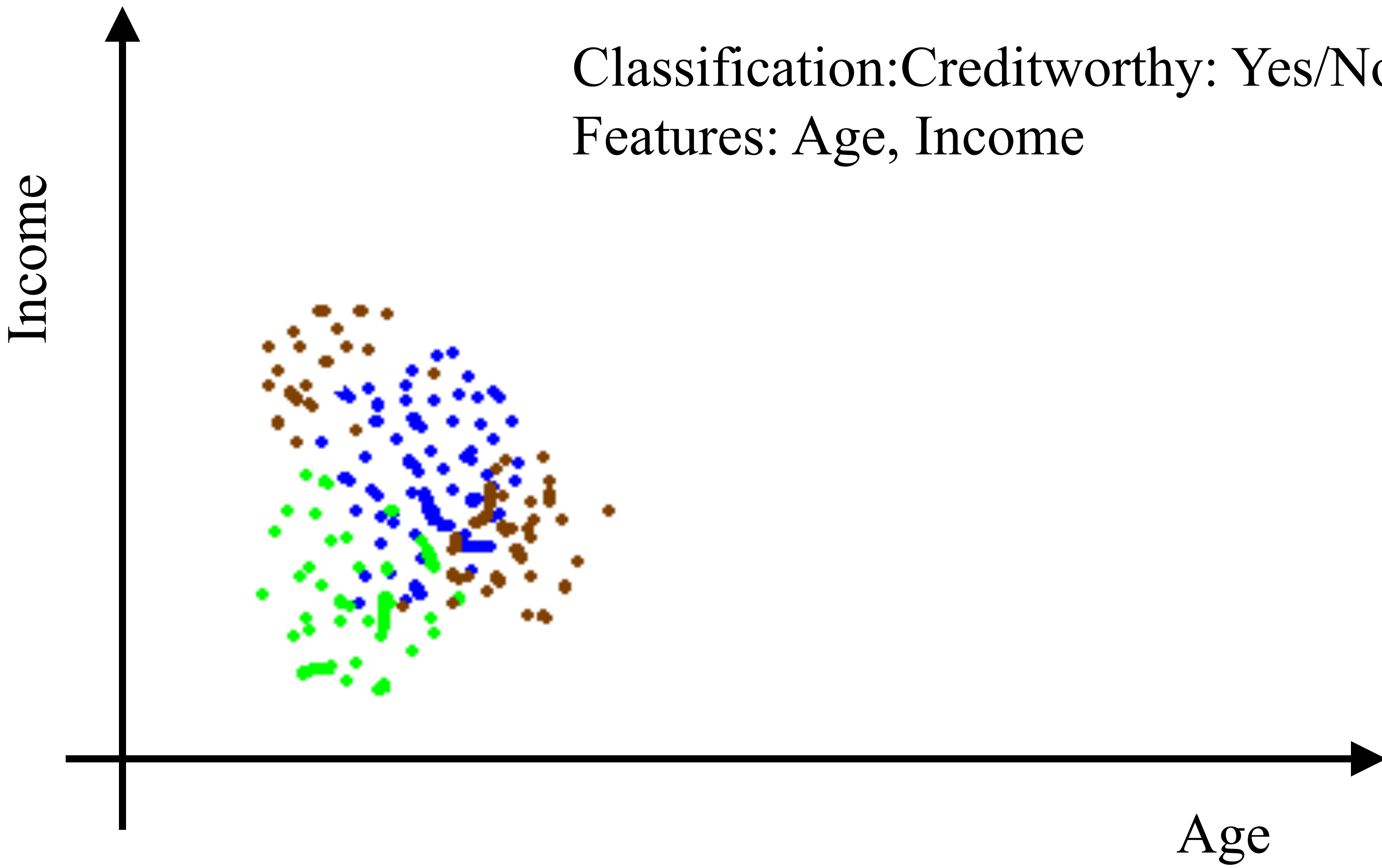
Pattern recognition

Classification:Creditworthy: Yes/No
Features: Age, Income



Pattern recognition

Classification:Creditworthy: Yes/No
Features: Age, Income

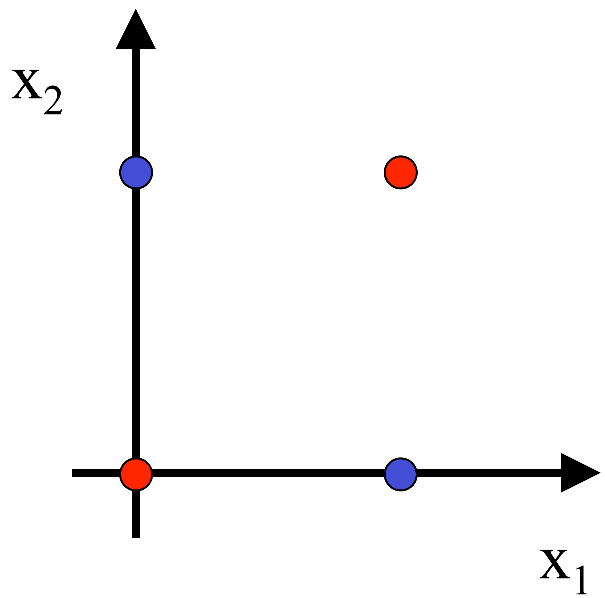


The XOR-Problem

- Input-Output Pairs

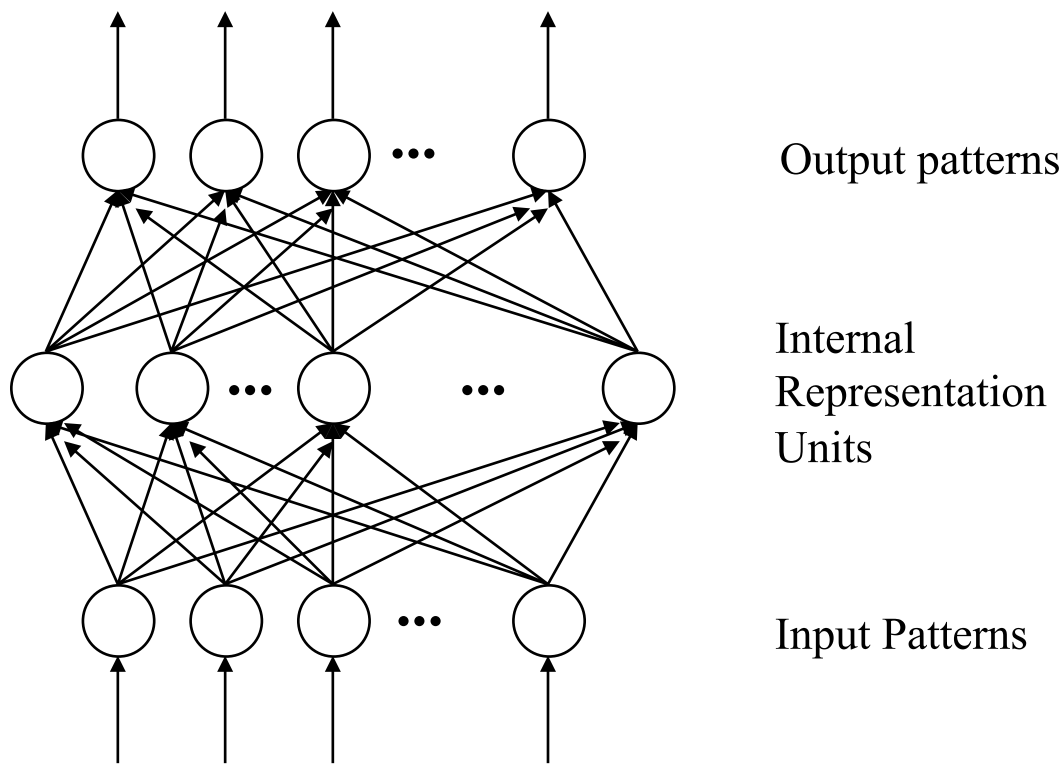
  0 0 -> 0
  0 1 -> 1
  1 0 -> 1
  1 1 -> 0

x_1 x_2



The Multi-Layer Perceptron

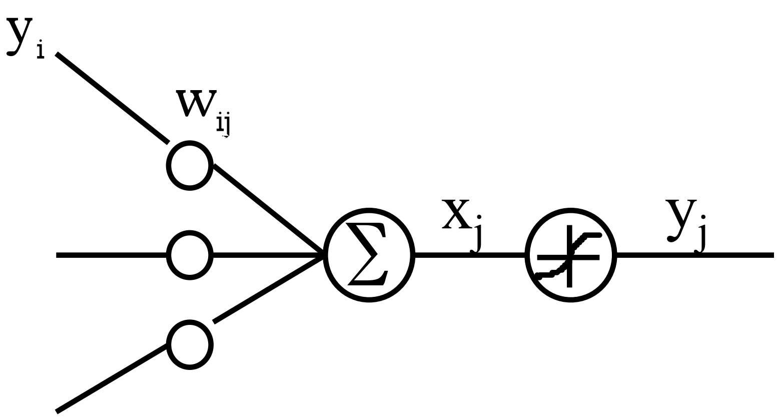
- Many interconnected simple processing elements:



Training the MLP by Error-Back-Propagation

- Choose random initial weights
- Apply input, get some output
- Compare output to *desired* output and compute error
- Back-propagate error through net and compute $E / \ w_{ij}$, the contribution of each weight to the overall error
- Adjust weights slightly to reduce error

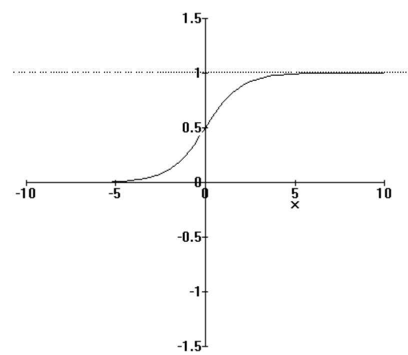
Back-Propagation of Error

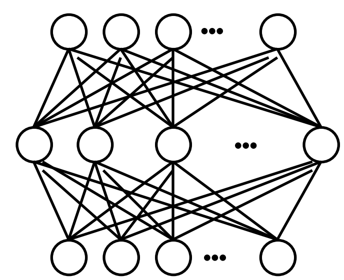


$$x_j \quad \sum_i y_i w_{ij}$$

$$y_j \quad \frac{1}{1 \quad e^{\ x_j}}$$

sigmoid





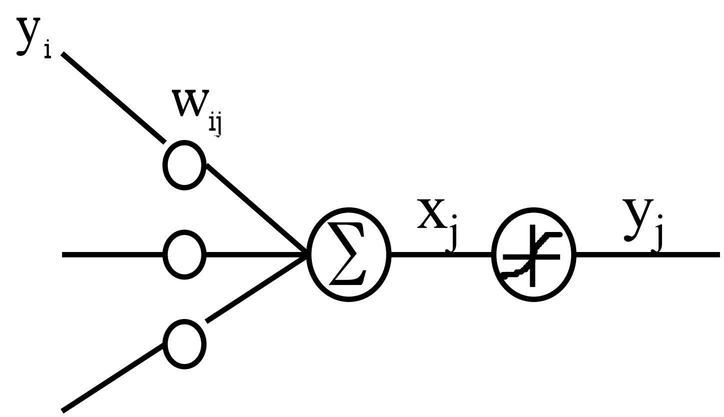
<u>Backpropagation of error:</u>

$$E \quad \frac{1}{2}\sum_j (y_j \quad d_j)^2$$

Back-Propagation of Error

$$E \quad \frac{1}{2}\sum_j (y_j \quad d_j)^2 \qquad y_j \quad \frac{1}{1 \quad e^{\;x_j}} \qquad x_j \quad \sum_i y_i w_{ij}$$

1). $$\frac{E}{y_j} \quad y_j \quad d_j$$



2). $$\frac{E}{x_j} \quad \frac{E}{y_j}\frac{y_j}{x_j} \quad \frac{E}{y_j} y_j [1 \quad y_j]$$

3). $$\frac{E}{w_{ij}} \quad \frac{E}{x_j}\frac{x_j}{w_{ij}} \quad \frac{E}{x_j} \; y_i$$

4). $$\frac{E}{y_i} \quad \sum_j \frac{E}{x_j}\frac{x_j}{y_i} \quad \sum_j \frac{E}{x_j} \; w_{ij}$$

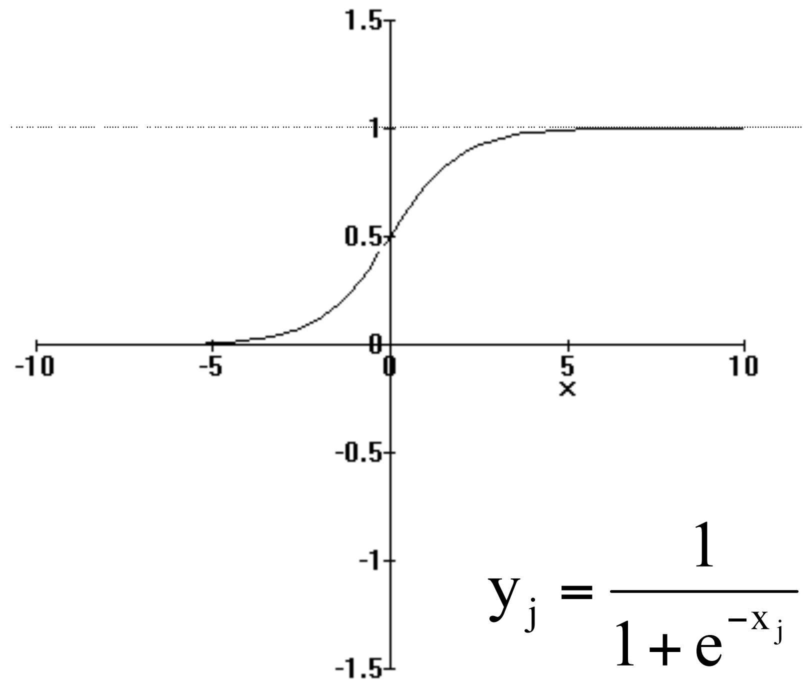
Derivative dy/dx

$$\frac{y_j}{x_j} \quad (\quad 1)(1 \quad e^{x_j})^{2} \quad (\quad 1)e^{x_j}$$

$$\frac{e^{x_j} \quad 1}{(1 \quad e^{x_j}) \; (1 \quad e^{x_j})}$$

$$\frac{e^{x_j}}{1 \quad e^{x_j}} \quad y_j$$

$$y_j(1 \quad y_j)$$



$$y_j = \frac{1}{1+e^{-x_j}}$$

Statistical Interpretation of MLP's

- What is the output of an MLP?
- Output represents a Posteriori Propabilities P(w|x)
- Assumptions:
  - Training Targets: 1, 0
  - Output Error Function: Mean Squared Error
  - Other Functions Possible

Statistical Interpretation of MLP's

- What is the output of an MLP?
- Using MSE, for two class problem, $c_1 \;\hat{}\; 1,\; c_2 \;\hat{}\; 0$

$$E \quad \frac{1}{N} \sum_{x \; c_1} [f(x) \quad 1]^2 \quad \sum_{x \; c_2} [f(x)]^2$$

if N large and reflects prior distribution

$$E \quad \int (f(x) \quad 1)^2 \, p(x, c_1) dx \quad \int (f(x))^2 \, p(x, c_2) dx$$

Statistical Interpretation of MLP's

$$E \quad \int f^2[p(x_1 c_1) \quad p(x_1 c_2)]dx \quad 2\int f(x)p(x_1 c_1)dx$$

$$1 \quad \int p(x_1 c_1)dx$$

$$\int f^2(x)p(x)dx \quad 2\int f(x)p(x_1 c_1)dx \quad \int p(x_1 c_1)dx$$

$$\int [f^2(x)p(x) \quad 2f(x)p(x_1 c_1) \quad p(x_1 c_1)]dx$$

$$\int [f^2(x) \quad 2f(x)p(c_1 / x) \quad p(c_1 / x)]p(x)dx$$

$$\boxed{E \quad \int \underbrace{[f(x) \quad p(c_1 / x)]}_{\text{net aprox. posterior}}{}^2 p(x)dx \quad \int p^2(c_1 / x)p(x)dx \quad P(c_1)}$$

What does the sigmoid do?

$$y_j \qquad \frac{1}{1 \quad e^{\;x_j}}$$

$$e^{\;x_j} \qquad \frac{1}{y_j} \quad 1$$

$$e^{x_j} \qquad \frac{y_j}{1 \quad y_j}$$

$$x_j \qquad \log[\frac{y_j}{1 \quad y_j}]$$

What does the sigmoid do?

If $y_j \quad p(q/x)$ **then**

$$x_j \quad \log\left[\frac{p(q/x)}{p(\bar{q}/x)}\right]$$

$$\underbrace{\log\left[\frac{p(x/q)}{p(x/\bar{q})}\right]}_{\text{inputs}} \quad \underbrace{\log\left[\frac{p(q)}{p(\bar{q})}\right]}_{\text{Bias}}$$

log. likelihood ratio or "odds"

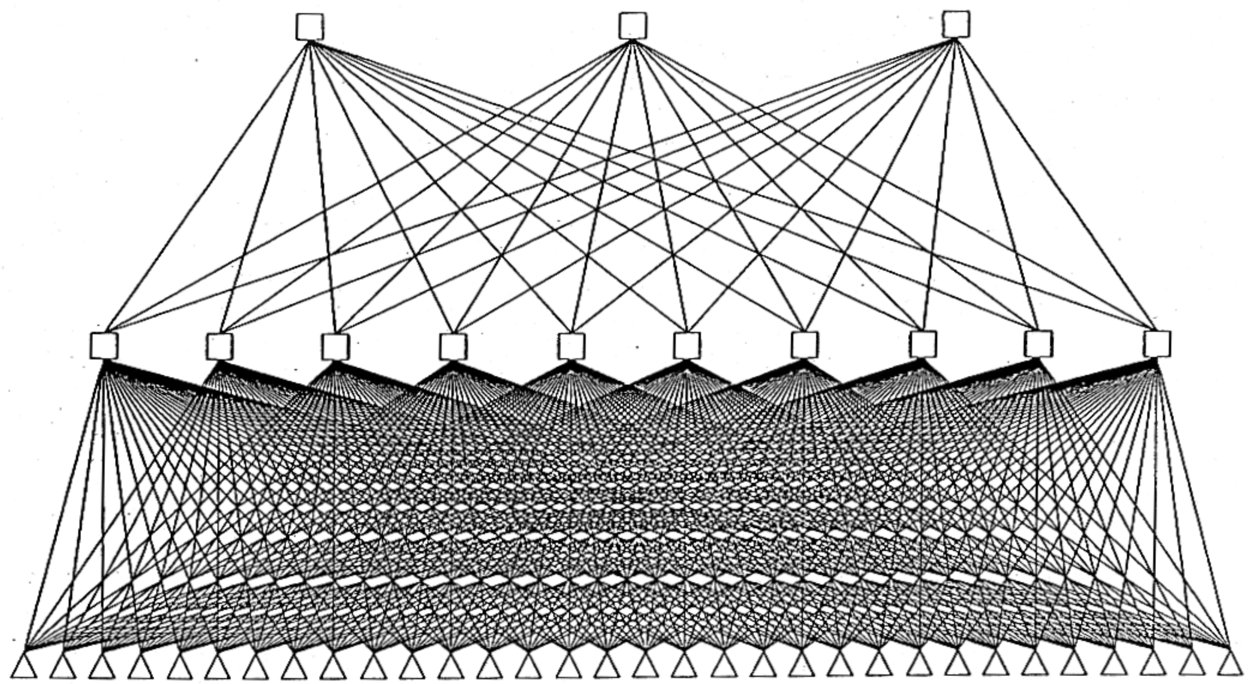
Use of Neural Nets

- Classification
  - Output Units Represent Classes
  - 1 of N bits is Switched on
  - Network trained to Produce Correct Output Pattern
  - Output Values are A Posteriori Probabilities
- Function Approximation
  - Output Units Represent Function Values
  - Network Learns to Behave like Function
  - Often Sigmoid Removed on Output
- Coding
  - Train Net to Reproduce Input Pattern at Output
  - Hidden Units Learn Code
  - Example: 4-2-4 Encoder

Using Neural Nets

- Classification
- Prediction
- Function Approximation
- Continuous Mapping
- Pattern Completion
- Coding

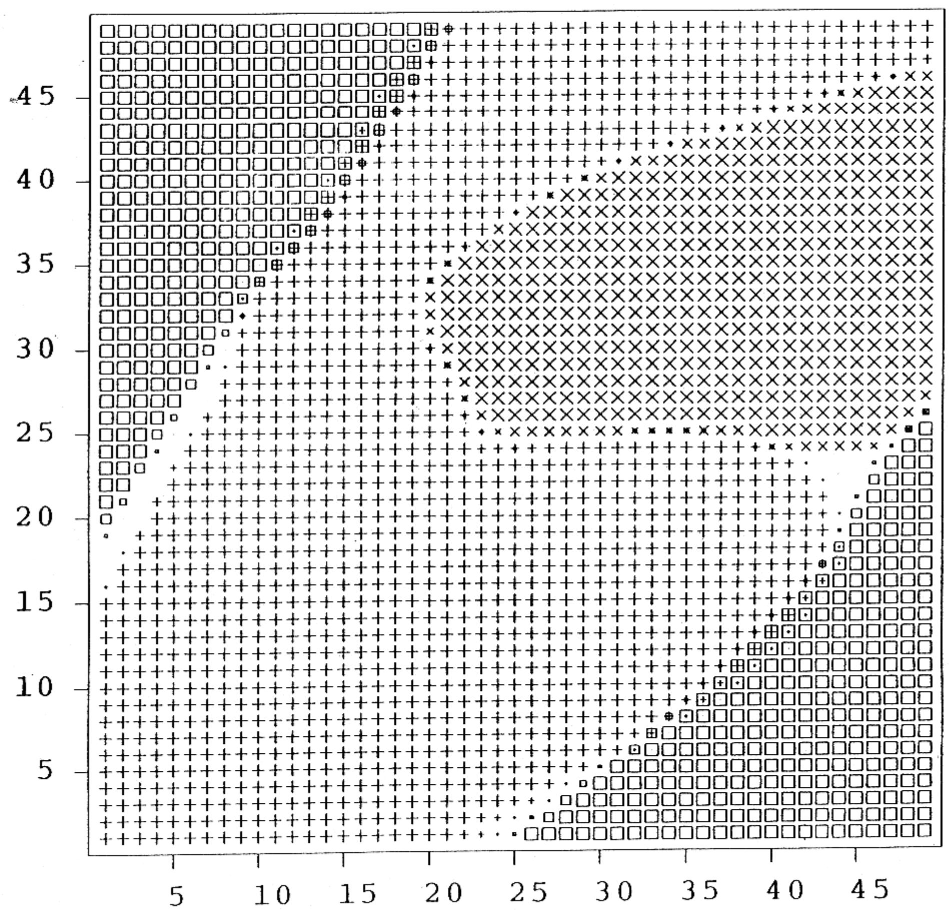
Neural Network for Motor Testing

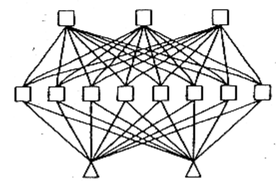


Neural Network for Motor Testing (32-10-3)

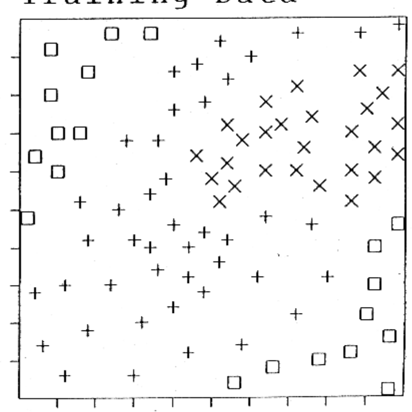
Neural Network for Motor Testing

Output Unit Responses after 13000 training passes

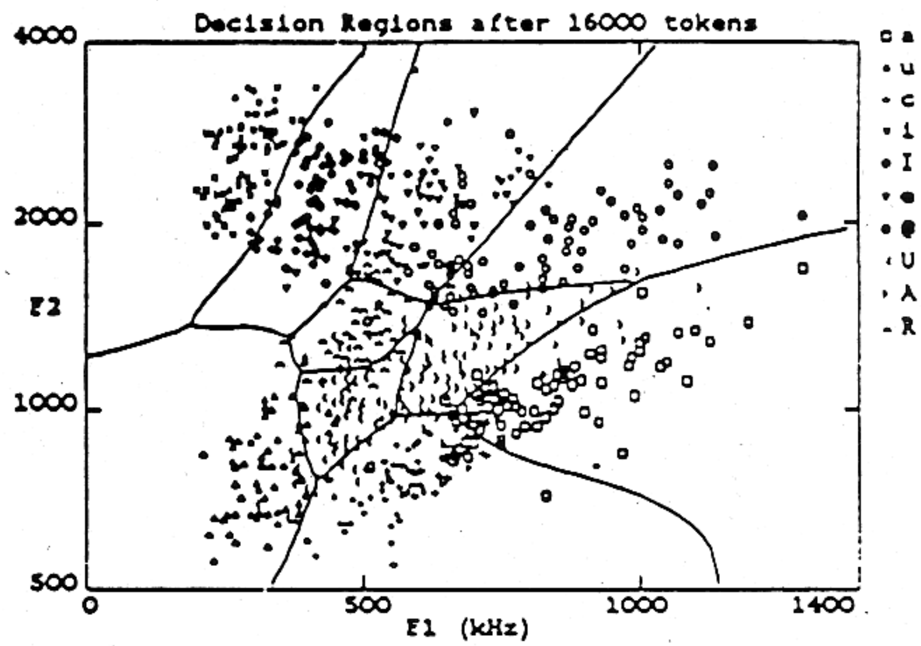




Training Data



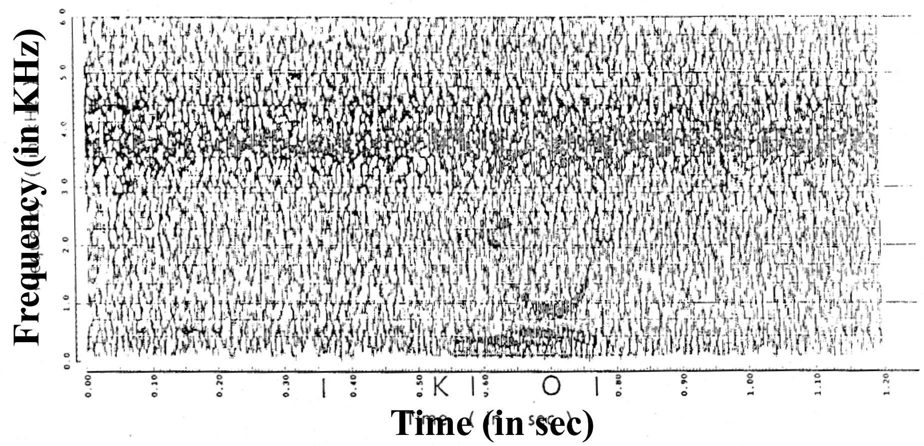
Lippmann, Vowel Classification

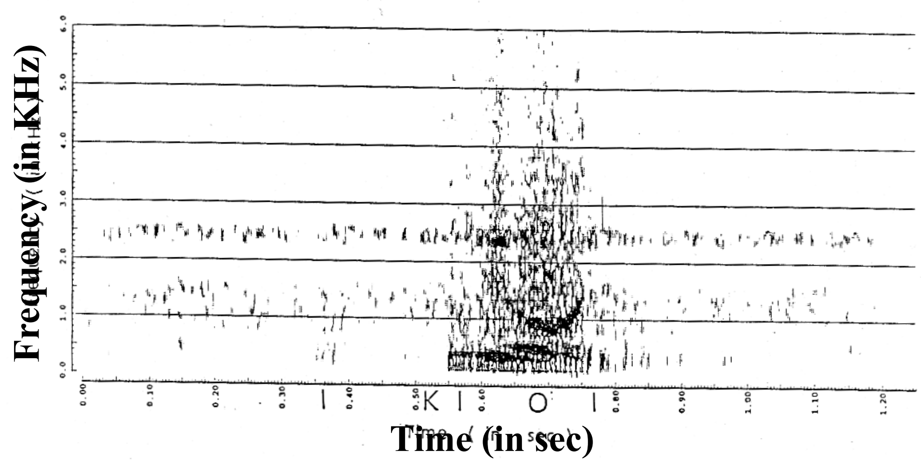


Decision regions formed after 16000 training examples
from Peterson and Barney's data. Data samples are also shown.
The Legend shows vowels in Arpabet notation.

Spectrograms of the Network Input and Output -Example 1-

(Speaker: Training, Word: Training, Noise: Training)





Design Criteria

- Recognition Error Rate
- Training Time
- Recognition Time
- Memory Requirements
- Training Complexity
- Ease of Implementation
- Ease of Adaptation

Network Specification

What parameters are typically chosen by the network designer:

- Net Topology
- Node Characteristics
- Learning Rule
- Objective Function
- (Initial) Weights
- Learning Parameters

Neural Nets - Design Problems

- Local Minima
- Speed of Learning
- Architecture must be selected
- Choice of Feature Representation
- Scaling
- Systems, Modularity
- Treatment of Temporal Features and Sequences

Neural Nets - Modeling

- Neural Nets are
  - Non-Linear Classifiers
  - Approximate Posterior Probabilities
  - Non-Parametric Training
- The Problem with Time
  - How to Consider Context
  - How to Operate Shift-Invariantly
  - How to Process Pattern Sequences

Applications

- Space Robot*
- Autonomous Navigation*
- Speech Recognition and Understanding*
- Natural Language Processing*
- Music*
- Gesture Recognition
- Lip Reading
- Face Recognition
- Household Robots
- Signal Processing
- Banking, Bond Rating...
- Sonar
- etc .....

Neural Models

- Back-Propagation
- Bolzman Machines
- Decision Tree Classifiers
- Feature Map Classifiers
- Learning Vector Quantizier (LVQ, LVQ2)
- High Order Networks
- Radial Basis Functions
- Modified Nearest Neighbor
- ART
- etc.

Time Varying Patterns

- Detect B in Context A
  - AAABAAA ---> 1
  - AAABCAA ---> 0
- Detect B Independent of Point in Time
  - AAAABAAAAAA ---> 1
  - AAAAAAABAAA ---> 1
  - AAAAAAAAAAA ----> 0
- Detect Sequence ABC
  - AAAAAABBCCCC ---> 1
  - ABBBBBBCCC ---> 1
  - CCAAABB ---> 0

Advanced Neural Models

- Time-Delay Neural Networks (Waibel)
- Recurrent Nets (Elman, Jordan)
- Higher Order Nets
- Modular System Construction
- Adaptive Architectures

Time-Delay Neural Networks

- Multilayer Neural Network - Nonlinear Classifier
- Time-Delays on Connections
  - At Input Layer
  - At all Layers
- Shift-Invariant Learning
  - All Units Learn to Detect Patterns Independent of Location in Time
  - No Presegmentation or Prealignment Necessary
  - Approach: Weight Sharing