Plan for Today

- Application Fields of ANNs
  - Classification
  - Prediction
  - Function Approximation
- Radial Basis Function Nets (RBF Nets)
  - Model
  - Training
- Recurrent MLP
  - Elman Nets
  - Jordan Nets

Application Fields of ANNs

Classification

given: set of training patterns (input / output)

\[ \vec{x}_i, \vec{y}_i \]

output = label
(e.g. class A, class B, ...)

training patterns
(known)

weights
(learnt)

input
(unknown)

output
(guessed)

parameters

\[ f(\vec{x}; \vec{x}_1, \vec{y}_1, \ldots, \vec{x}_m, \vec{y}_m, w_1, \ldots, w_n) \rightarrow \vec{y} \]

phase I:
train network

phase II:
apply network to unknown inputs for classification

Prediction of Time Series

time series \( x_1, x_2, x_3, \ldots \) (e.g. temperatures, exchange rates, ...)

task: given a subset of historical data, predict the future

\[ f(x_{t-k}, x_{t-k+1}, \ldots, x_t, w_1, \ldots, w_n) \rightarrow x_{t+T} \]

phase I:
train network

phase II:
apply network to historical inputs for predicting unknown outputs

training patterns:
historical data where true output is known:
error per pattern = \( (\hat{x}_{t+T} - x_{t+T})^2 \)
**Function Approximation** (the general case)

- Task: given training patterns (input/output), approximate unknown function
- Should give outputs close to true unknown function for arbitrary inputs
  - Values between training patterns are **interpolated**
  - Values outside convex hull of training patterns are **extrapolated**

**Radial Basis Function Nets (RBF Nets)**

**Definition:**
A function \( f : \mathbb{R}^n \rightarrow \mathbb{R} \) is termed radial basis function (RBF function) iff
\[
\forall x \in \mathbb{R}^n : f(x; c) = \varphi(||x - c||).
\]

Examples:
- Gaussian: unbounded
- Epanechnikov: bounded
- Cosine: bounded

**Radial Basis Function Nets (RBF Nets)**

**Definition:**
A function \( f(x) \) is termed radial basis function net (RBF net) iff
\[
f(x) = w_1 \varphi(||x - c_1||) + w_2 \varphi(||x - c_2||) + \ldots + w_p \varphi(||x - c_p||).
\]

\( x \) : input training pattern
\( * \) : input pattern where output to be interpolated
\( \triangle \) : input pattern where output to be extrapolated

**Solution:**
We know that \( f(x_i) = y_i \) for \( i = 1, \ldots, N \) or equivalently
\[
\sum_{k=1}^{q} w_k \varphi(||x_i - c_k||) = y_i
\]

\( P_k \) : known value

\[
\Rightarrow \sum_{k=1}^{q} w_k \cdot p_{ik} = y_i \Rightarrow N \text{ linear equations with } q \text{ unknowns}
\]
### Radial Basis Function Nets (RBF Nets)

**Lecture 03**

**in matrix form:**  
\[ P \mathbf{w} = \mathbf{y} \]  
with  
\( P = (p_{ij}) \)  
and  \( P : N \times q, \mathbf{y} : N \times 1, \mathbf{w} : q \times 1 \).

**case**  \( N = q \):  
\( \mathbf{w} = \mathbf{P}^{-1} \mathbf{y} \)  
if  \( P \) has full rank.

**case**  \( N < q \):  
many solutions  
but of no practical relevance.

**case**  \( N > q \):  
\( \mathbf{w} = \mathbf{P}^+ \mathbf{y} \)  
where  \( \mathbf{P}^+ \) is Moore-Penrose pseudo inverse.

\[
\begin{align*}
\mathbf{P} \mathbf{w} &= \mathbf{y} \\
\mathbf{P}^\mathbf{P} \mathbf{w} &= \mathbf{P}^\mathbf{y} \\
(\mathbf{P}^\mathbf{P})^{-1} \mathbf{P} \mathbf{w} &= (\mathbf{P}^\mathbf{P})^{-1} \mathbf{P} \mathbf{y} \\
\text{unit matrix} &\quad \mathbf{P}^+ \\
\end{align*}
\]

**Complexity (naive):**  
\[
\begin{align*}
\mathbf{w} &= (\mathbf{P}^\mathbf{P})^{-1} \mathbf{P}^\mathbf{y} \\
\mathbf{P}^\mathbf{P}: N^2 q &\quad \text{inversion: } q^3 \\
\mathbf{P}^\mathbf{y}: qN &\quad \text{multiplication: } q^2
\end{align*}
\]

**Remark:**  
if  \( N \) large then inaccuracies for  \( \mathbf{P}^\mathbf{P} \) likely

\[ \Rightarrow \text{first analytic solution, then gradient descent starting from this solution} \]

- requires differentiable basis functions!

**Advantages:**
- additional training patterns  
  → only local adjustment of weights
- optimal weights determinable in polynomial time
- regions not supported by RBF net can be identified by zero outputs

**Disadvantages:**
- number of neurons increases exponentially with input dimension
- unable to extrapolate (since there are no centers and RBFs are local)
Recurrent MLPs

Lecture 03

Jordan nets (1986)
- context neuron:
  reads output from some neuron at step t and feeds value into net at step t+1

Elman nets (1990)
- Elman net = MLP + context neuron for each neuron output of MLP, context neurons fully connected to associated MLP layer

Training?
⇒ unfolding in time ("loop unrolling")
- identical MLPs serially connected (finitely often)
- results in a large MLP with many hidden (inner) layers
- backpropagation may take a long time
- but reasonable if most recent past more important than layers far away

Why using backpropagation?
⇒ use Evolutionary Algorithms directly on recurrent MLP!