BIOLOGICAL INSPIRATIONS

Some numbers...

The human brain contains about 10 billion nerve cells (neurons)
Each neuron is connected to the others through 10000 synapses

Properties of the brain

It can learn, reorganize itself from experience
It adapts to the environment
It is robust and fault tolerant
WHAT IS AN ARTIFICIAL NEURON?

Definition: Non linear, parameterized function with restricted output range

\[ y = f \left( w_0 + \sum_{i=1}^{n-1} w_i x_i \right) \]
**ACTIVATION FUNCTIONS**

Linear
\[ y = x \]

Logistic
\[ y = \frac{1}{1 + \exp(-x)} \]

Hyperbolic tangent
\[ y = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} \]
NEURAL NETWORKS

A mathematical model to solve engineering problems
Group of highly connected neurons to realize compositions of non-linear functions
Tasks
- Classification
- Discrimination
- Estimation

2 types of networks
- Feed forward Neural Networks
- Recurrent Neural Networks
FEED FORWARD NEURAL NETWORKS

The information is propagated from the inputs to the outputs.

Time has no role (NO cycle between outputs and inputs.)
Can have arbitrary topologies
Can model systems with internal states (dynamic ones)
Delays are associated to a specific weight
Training is more difficult
Performance may be problematic
Stable Outputs may be more difficult to evaluate
Unexpected behavior (oscillation, chaos, ...)

x1  x2
The procedure that consists in estimating the parameters of neurons so that the whole network can perform a specific task

2 types of learning
- The supervised learning
- The unsupervised learning

The Learning process (supervised)
- Present the network a number of inputs and their corresponding outputs
- See how closely the actual outputs match the desired ones
- Modify the parameters to better approximate the desired outputs
SUPERVISED LEARNING

The desired response of the neural network in function of particular inputs is well known. A “Professor” may provide examples and teach the neural network how to fulfill a certain task.
UNSUPERVISED LEARNING

Idea: group typical input data in function of resemblance criteria unknown a priori

Data clustering

No need of a professor

The network finds itself the correlations between the data

Examples of such networks:

Kohonen feature maps
Supervised networks are universal approximators (Non-recurrent networks)

Theorem: Any limited function can be approximated by a neural network with a finite number of hidden neurons to an arbitrary precision

Type of Approximators

Non-Linear approximators (NN): for a given precision, the number of parameters grows exponentially with the number of variables (polynomials)

Linear approximators: the number of parameters grows linearly with the number of variables
OTHER PROPERTIES

Adaptivity ✖
Adapt weights to environment and retrained easily +

Generalization ability ✖
May provide against lack of data +

Fault tolerance ✖
Graceful degradation of performances if damaged => The information is distributed within the entire net.
Not an approximation but a fitting problem

Regression function

Approximation of the regression function: Estimate the more probable value of $y_p$ for a given input $x$

Cost function:

$$J(w) = \frac{1}{2} \sum_{k=1}^{N} \left[ y_p(x^k) - g(x^k, w) \right]^2$$

Goal: Minimize the cost function by determining the right function $g$
EXAMPLE

Training sequence
Neural model (4 hidden neurons)
Process output

Training sequence
Neural model (8 hidden neurons)
Process output
Classification (Discrimination)

- Class objects in defined categories
- Rough decision
- Estimation of the probability for a certain object to belong to a specific class

Example: Data mining

Applications: Economy, speech and patterns recognition, sociology, etc.
Examples of handwritten postal codes drawn from a database available from the US Postal service
WHAT DO WE NEED TO USE NN?

- Determination of pertinent inputs
- Collection of data for the learning and testing phase of the neural network
- Finding the optimum number of hidden nodes
- Estimate the parameters (Learning)
- Evaluate the performances of the network

IF performances are not satisfactory then review all the precedent points
CLASSICAL NEURAL ARCHITECTURES

- Perceptron
- Multi-Layer Perceptron
- Radial Basis Function (RBF)
- Kohonen Features maps
- Other architectures

An example: Shared weights neural networks
Rosenblatt (1962)

Linear separation

Inputs: Vector of real values

Outputs: 1 or -1

\[ y = \text{sign}(v) \]
\[ v = c_0 + c_1 x_1 + c_2 x_2 \]
The perceptron algorithm converges if examples are linearly separable
MULTI-LAYER PERCEPTRON

One or more hidden layers

Sigmoid activation functions

Output layer

2nd hidden layer

1st hidden layer

Input data
Back-propagation algorithm

\[ net_j = w_{j0} + \sum_{i}^{n} w_{ji} o_i \]

\[ o_j = f_j (net_j) \]

\[ \Delta w_{ji} = -\alpha \frac{\partial E}{\partial w_{ji}} = -\alpha \frac{\partial E}{\partial net_j} \frac{\partial net_j}{\partial w_{ji}} = \alpha \delta_j o_i \]

\[ \delta_j = -\frac{\partial E}{\partial o_j} \frac{\partial o_j}{\partial net_j} = -\frac{\partial E}{\partial o_j} f'(net_j) \]

\[ E = \frac{1}{2} (t_j - o_j)^2 \Rightarrow \frac{\partial E}{\partial o_j} = -(t_j - o_j) \]

\[ \delta_j = (t_j - o_j) f'(net_j) \]

Credit assignment

\[ \delta_j = -\frac{\partial E}{\partial net_j} \]

If the jth node is an output unit
\[
\frac{\partial E}{\partial o_j} = \sum_{k}^{\kappa} \frac{\partial E}{\partial net_\kappa} \frac{\partial net_\kappa}{\partial o_j} = -\sum_{k}^{\kappa} \delta_k w_{kj}
\]

\[
\delta_j = f'_j (net_j) \sum_{k}^{\kappa} \delta_k w_{kj}
\]

\[
\Delta w_{ji}(t) = \alpha \delta_j(t) o_i(t) + \gamma \Delta w_{ji}(t - 1)
\]

\[
w_{ji}(t) = w_{ji}(t - 1) + \Delta w_{ji}(t)
\]

Momentum term to smooth
The weight changes over time
### Different Non Linearly Separable Problems

<table>
<thead>
<tr>
<th>Structure</th>
<th>Types of Decision Regions</th>
<th>Exclusive-OR Problem</th>
<th>Classes with Meshed regions</th>
<th>Most General Region Shapes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-Layer</td>
<td>Half Plane Bounded By Hyperplane</td>
<td>A</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>Two-Layer</td>
<td>Convex Open Or Closed Regions</td>
<td>A</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>Three-Layer</td>
<td>Arbitrary (Complexity Limited by No. of Nodes)</td>
<td>A</td>
<td>B</td>
<td></td>
</tr>
</tbody>
</table>

*Neural Networks – An Introduction Dr. Andrew Hunter*
RADIAL BASIS FUNCTIONS (RBFS)

Features

One hidden layer

The activation of a hidden unit is determined by the distance between the input vector and a prototype vector
RBF hidden layer units have a receptive field which has a centre.

Generally, the hidden unit function is Gaussian.

The output Layer is linear.

Realized function

\[
s(x) = \sum_{j=1}^{K} W_j \Phi(\|x - c_j\|)
\]

\[
\Phi(\|x - c_j\|) = \exp\left(-\left(\frac{\|x - c_j\|}{\sigma_j}\right)^2\right)
\]
The training is performed by deciding on:

- How many hidden nodes there should be
- The centers and the sharpness of the Gaussians

2 steps:

In the 1st stage, the input data set is used to determine the parameters of the basis functions.

In the 2nd stage, functions are kept fixed while the second layer weights are estimated (Simple BP algorithm like for MLPs).
MLPS VERSUS RBFS

**Classification**
- MLPs separate classes via hyperplanes
- RBFs separate classes via hyperspheres

**Learning**
- MLPs use distributed learning
- RBFs use localized learning
- RBFs train faster

**Structure**
- MLPs have one or more hidden layers
- RBFs have only one layer
- RBFs require more hidden neurons \(\Rightarrow\) curse of dimensionality
The purpose of SOM is to map a multidimensional input space onto a topology preserving map of neurons. Preserve a topological so that neighboring neurons respond to « similar » input patterns. The topological structure is often a 2 or 3 dimensional space. Each neuron is assigned a weight vector with the same dimensionality of the input space. Input patterns are compared to each weight vector and the closest wins (Euclidean Distance).
The activation of the neuron is spread in its direct neighborhood => neighbors become sensitive to the same input patterns

Block distance

The size of the neighborhood is initially large but reduce over time => Specialization of the network
During training, the “winner” neuron and its neighborhood adapts to make their weight vector more similar to the input pattern that caused the activation. The neurons are moved closer to the input pattern. The magnitude of the adaptation is controlled via a learning parameter which decays over time.
SHARED WEIGHTS NEURAL NETWORKS: TIME DELAY NEURAL NETWORKS (TDNNS)

Introduced by Waibel in 1989

Properties

Local, shift invariant feature extraction
Notion of receptive fields combining local information into more abstract patterns at a higher level
Weight sharing concept (All neurons in a feature share the same weights)
All neurons detect the same feature but in different position

Principal Applications

Speech recognition
Image analysis
**TDNNS (CONT’D)**

Objects recognition in an image

- Each hidden unit receives inputs only from a small region of the input space: receptive field

- Shared weights for all receptive fields => translation invariance in the response of the network
Advantages

Reduced number of weights +

Require fewer examples in the training set ×

Faster learning ×

Invariance under time or space translation +

Faster execution of the net (in comparison of full connected MLP) +
NEURAL NETWORKS (APPLICATIONS)

- Security
- Face recognition
- Time series prediction
- Process identification
- Process control
- Optical character recognition
- Adaptative filtering
- Etc...
CONCLUSION ON NEURAL NETWORKS

- Neural networks are utilized as statistical tools
  - Adjust non linear functions to fulfill a task
  - Need of multiple and representative examples but fewer than in other methods

- Neural networks enable to model complex static phenomena as well as dynamic ones

- NN are good classifiers and are used in security tasks.

- The use of NN needs a good comprehension of the problem
THANK YOU!