# LECTURE 42: NEURAL NETWORKS

- Objectives:
  - Comparison to HMM States (Neurons)
  - Nonlinearities
  - Multi-layer Perceptron
  - Recurrent Networks

This lecture uses material from:

J. Deller, et. al., *Discrete-Time Processing of Speech Signals*, MacMillan Publishing Co., ISBN: 0-7803-5386-2, 2000.

and the course textbook:

X. Huang, A. Acero, and H.W. Hon, *Spoken Language Processing - A Guide to Theory, Algorithm, and System Development*, Prentice Hall, Upper Saddle River, New Jersey, USA, ISBN: 0-13-022616-5, 2001.

# LECTURE 42: NEURAL NETWORKS

- Objectives:
  - Comparison to HMM States (Neurons)
  - Nonlinearities
  - Multi-layer Perceptron
  - Recurrent Networks

This lecture uses material from:

J. Deller, et. al., *Discrete-Time Processing of Speech Signals*, MacMillan Publishing Co., ISBN: 0-7803-5386-2, 2000.

and the course textbook:

X. Huang, A. Acero, and H.W. Hon, *Spoken*

*Language Processing - A Guide to Theory, Algorithm, and System Development*, Prentice Hall, Upper Saddle River, New Jersey, USA, ISBN: 0-13-022616-5, 2001.

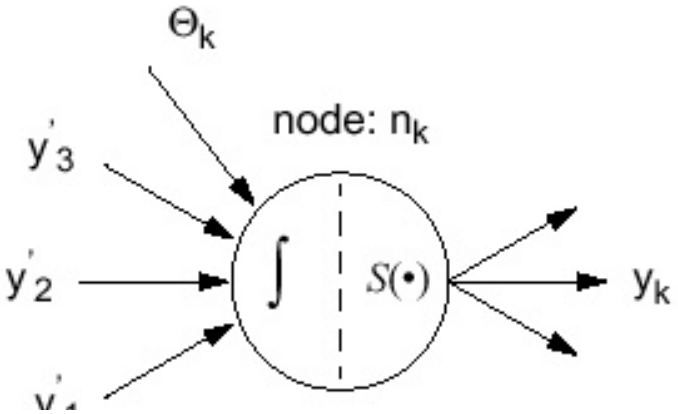
# THE ARTIFICIAL NEURAL NETWORK (ANN)

- Premise: complex computational operations can be implemented by massive integration of individual components
- Topology and interconnections are key: in many ANN systems, spatial relationships between nodes have some physical relevance
- Properties of large-scale systems: ANNs also reflect a growing body of theory stating that large-scale systems built from a small unit need not simply mirror properties of a smaller system (contrast fractals and chaotic systems with digital filters)

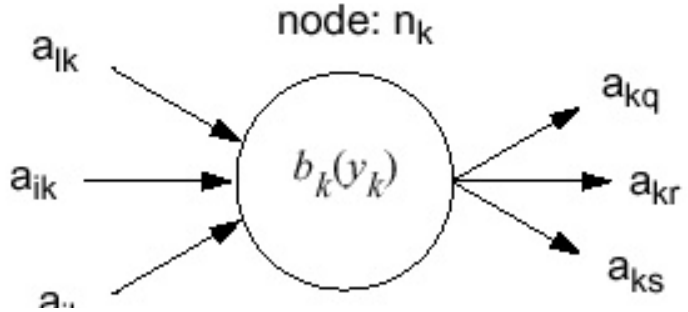
Why Artificial Neural Networks?

- Important physical observations:
  - The human central nervous system contains $10^{11}$ — $10^{14}$ nerve cells, each of which interacts with $10^3$ — $10^4$ other neurons
  - Inputs may be excitatory (promote firing) or inhibitory

**The Artificial Neuron — Nonlinear**




**The HMM State — Linear**



**vector input** **scalar output**

$$y_k \equiv S\left(\sum_{n=1}^{N} \boldsymbol{w}_{ki} \boldsymbol{y}'_i - \theta_k\right)$$

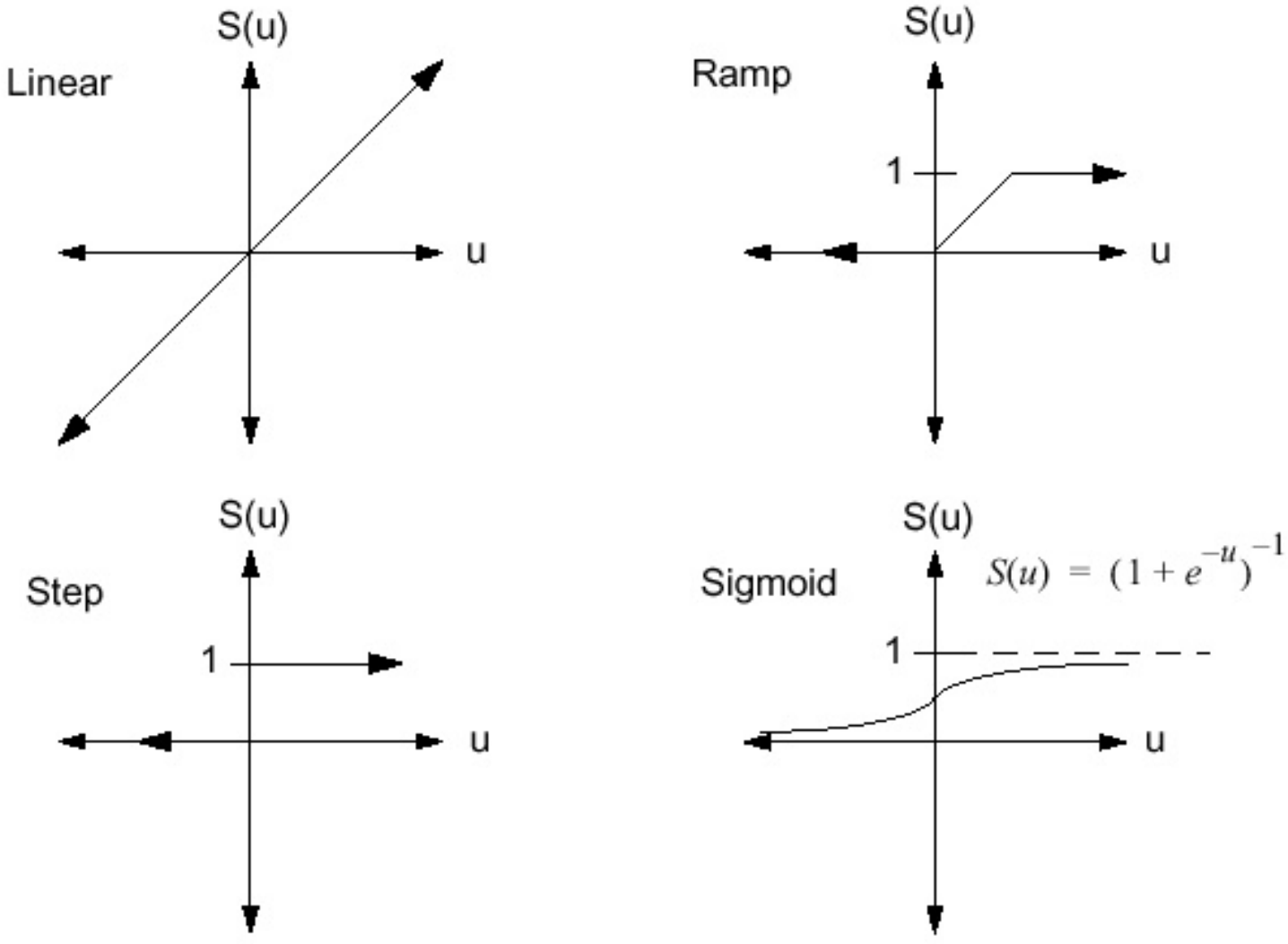
$$\alpha(y_1^{t+1}, j) = \alpha(y_1^t, i)\, a(j|i)\, b(y(t+1)|j)$$

# TYPICAL THRESHOLDING FUNCTIONS - A KEY DIFFERENCE

The input to the thresholding function is a weighted sum of the inputs:

$$u_k \equiv \boldsymbol{w}_k^T \boldsymbol{y}'$$

The output is typically defined by a nonlinear function:




Sometimes a bias is introduced into the threshold function:

$$y_k \equiv S(\boldsymbol{w}_k^T \boldsymbol{y}' - \theta_k) = S(u_k - \theta_k)$$

This can be represented as an extra input whose value is always -1:

This can be represented as an extra input whose value is always $-1$:

$$y'_{N+1} = -1 \qquad w_{k,N+1} = \theta_k$$

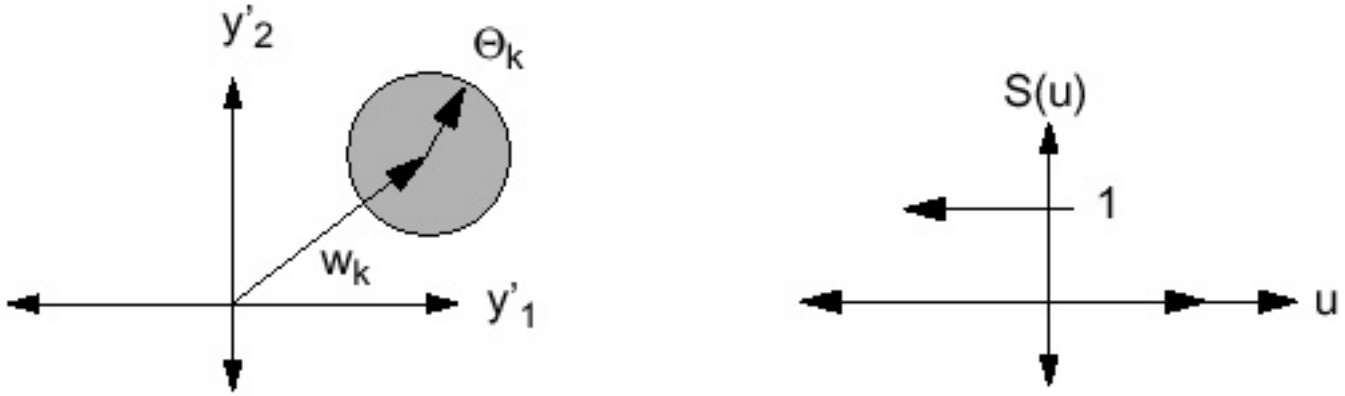
# RADIAL BASIS FUNCTIONS

Another popular formulation involves the use of a Euclidean distance:

$$y_k = S\left(\sqrt{\sum_{i=1}^{N}(w_{ik} - y'_i)^2} - \theta_k\right) = S\left(\left\|\boldsymbol{w}_k - \boldsymbol{y}'\right\|_2 - \theta_k\right)$$

Note the parallel to a continuous distribution HMM.

This approach has a simple geometric interpretation:




Another popular variant of this design is to use a Gaussian nonlinearity:

$$S(u) = e^{-u^2}$$

What types of problems are such networks useful for?

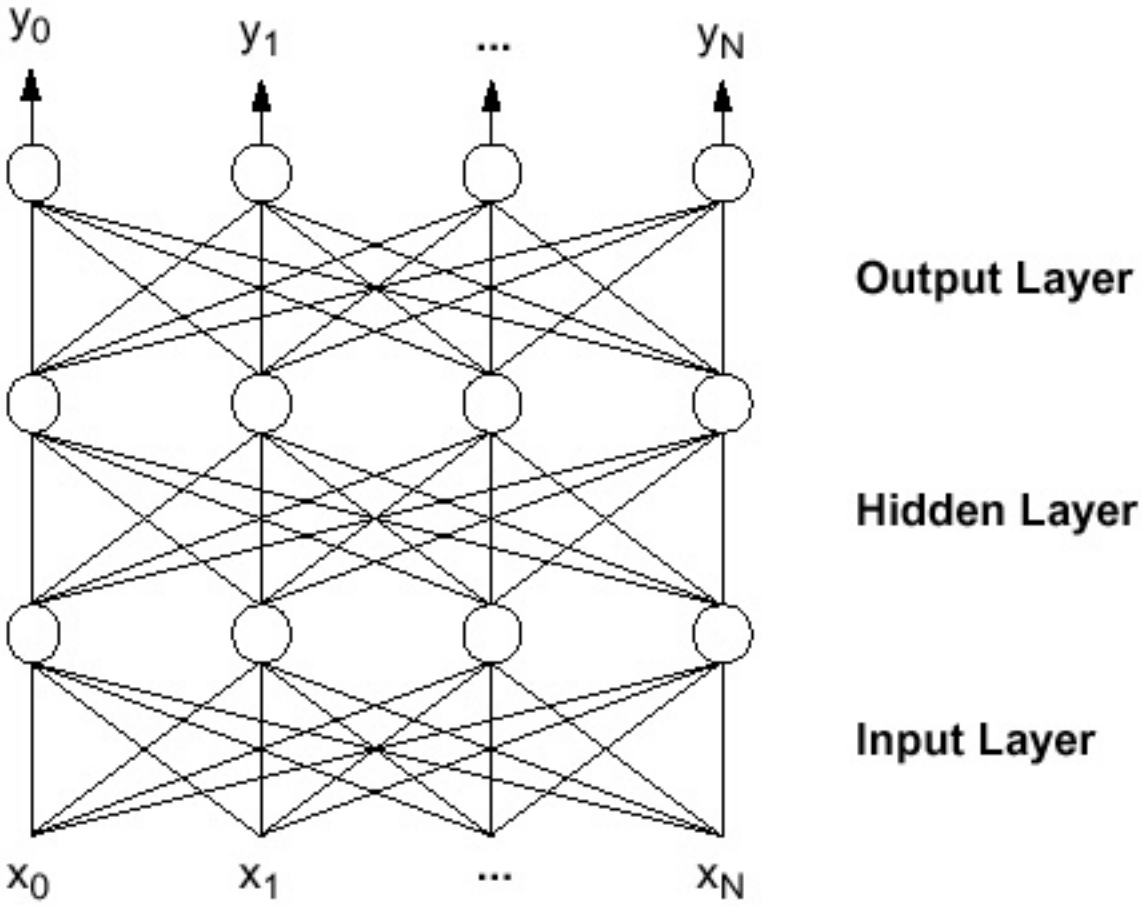
- pattern classification (N-way choice; vector quantization)
- associative memory (generate an output from a noisy input; character recognition)
- feature extraction (similarity transformations; dimensionality reduction)

We will focus on multilayer perceptrons in our studies. These have been shown to be quite useful for a wide range of problems.

# MULTI-LAYER PERCEPTRONS

This architecture has the following characteristics:

- Network segregated into layers: $N_i$ cells per layer, L layers
- feedforward, or nonrecurrent, network (no feedback from the output of a node to the input of a node)



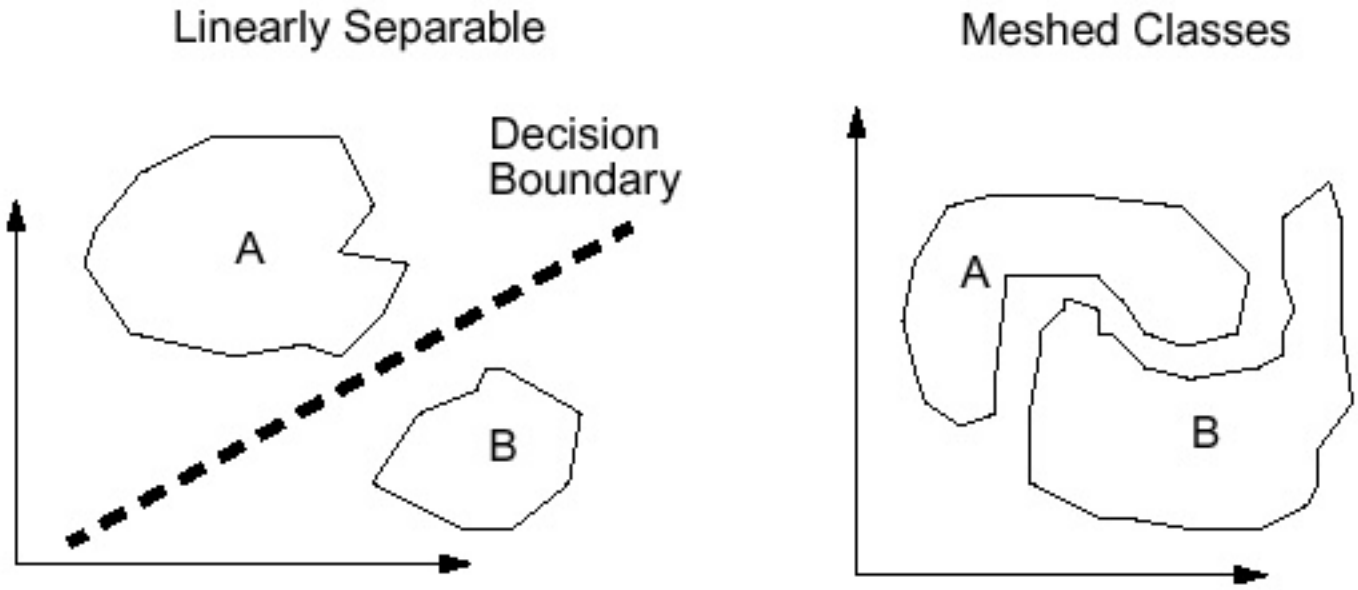

An alternate formulation of such a net is known as the learning vector quantizer (LVQ) — to be discussed later.

The MLP network, not surprisingly, uses a supervised learning algorithm. The network is presented the input and the corresponding output, and must learn the optimal weights of the coefficients to minimize the difference between these two.

The LVQ network uses unsupervised learning — the network adjusts itself automatically to the input data, thereby clustering the data (learning the boundaries representing a segregation of the data). LVQ is popular because it supports discriminative training.

# WHY ARTIFICIAL NEURAL NETWORKS FOR SPEECH?

- An ability to separate classes that are not linearly separable:




A three-layer perceptron is required to determine arbitrarily-shaped decision regions.

- Nonlinear statistical models

  The ANN is capable of modeling arbitrarily complex probability distributions, much like the difference between VQ and continuous distributions in HMM.

- Context-sensitive statistics

  Again, the ANN can learn complex statistical dependencies provided there are enough degress of freedom in the system.

**Why not Artificial Neural Networks? (The Price We Pay...)**

- Difficult to deal with patterns of unequal length

- Difficult to deal with patterns of unequal length

- Temporal relationships not explicitly modeled

And, of course, both of these are extremely important to the speech recognition problem.

# MLP TRAINING: BACK PROPAGATION

By incorporating a nonlinear transfer function that is differentiable, we can derive an iterative gradient descent training algorithm for a multi-layer perceptron (MLP). This algorithm is known as **back propagation**:

**ALGORITHM 4.1: *THE BACK PROPAGATION ALGORITHM***

**Step 1:** Initialization: Set $t = 0$ and choose initial weight matrices $\mathbf{W}$ for each layer. Let's denote $w_{ij}^k(t)$ as the weighting coefficients connecting $i^{th}$ input node in layer $k-1$ and $j^{th}$ output node in layer $k$ at time $t$.

**Step 2:** Forward Propagation: Compute the values in each node from input layer to output layer in a propagating fashion, for $k = 1$ to $K$

$$v_j^k = sigmoid(w_{0j}(t) + \sum_{i=1}^{N} w_{ij}^k(t) v_i^{k-1}) \quad \forall j \tag{4.72}$$

where $sigmoid(x) = \dfrac{1}{1+e^{-x}}$ and $v_j^k$ is denoted as the $j^{th}$ node in the $k^{th}$ layer

**Step 3:** Back Propagation: Update the weights matrix for each layer from output layer to input layer according to:

$$\overline{w}_{ij}^k(t+1) = w_{ij}^k(t) - \alpha \frac{\partial E}{\partial w_{ij}^k(t)} \tag{4.73}$$

where $E = \sum_{i=1}^{s} \| y_i - o_i \|^2$ and $(y_1, y_2, \ldots y_s)$ is the computed output vector in Step 2.

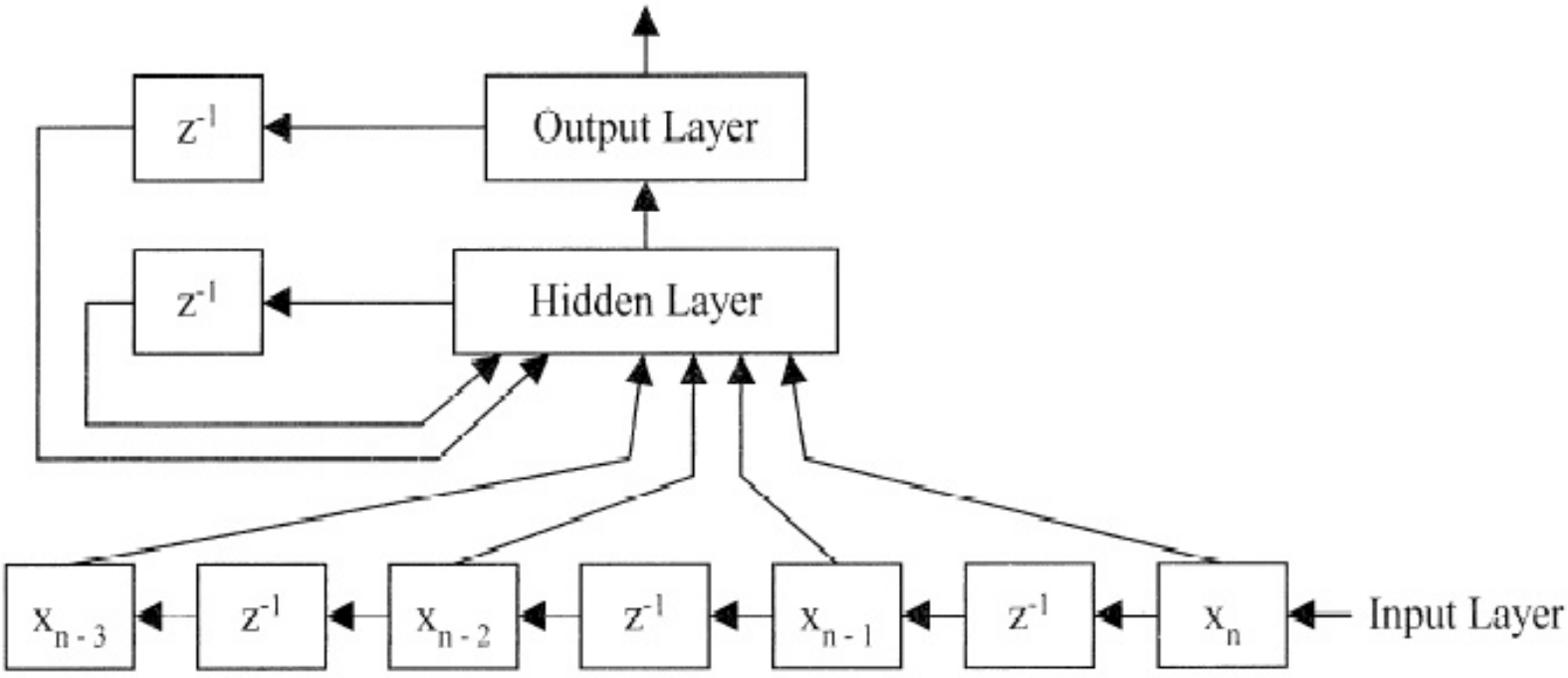
$\alpha$ is referred to as the learning rate and has to be small enough to guarantee convergence. One popular choice is $1/(t+1)$.

**Step 4:** Iteration: Let $t = t+1$. Repeat Steps 2 and 3 until some convergence condition is met.

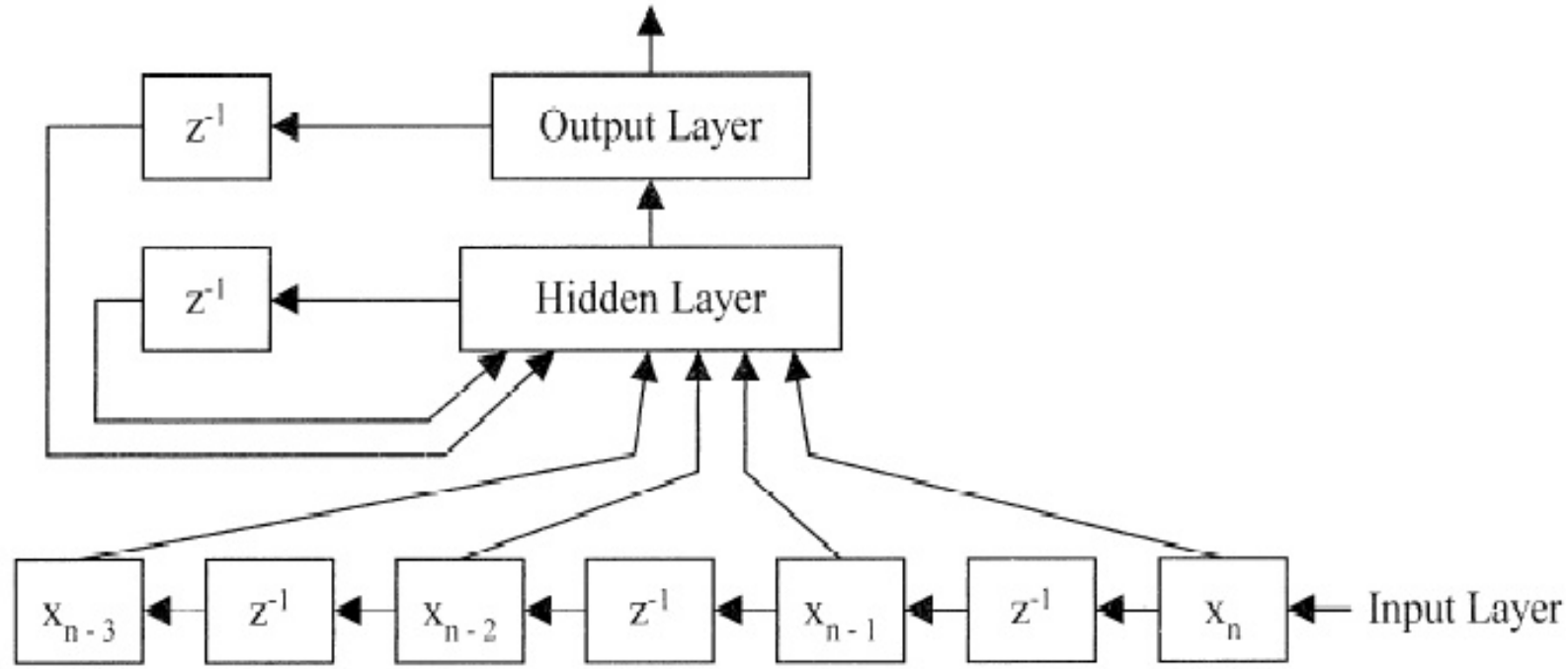
The MLP network has been the most popular architecture for speech processing applications due to the existence of robust training algorithms and its powerful classification properties.

# RECURRENT NETWORKS: TOWARDS TIME SYNCHRONOUS DECODING

To incorporate time synchronous behavior into a neural network, we need some sort of feedback looop. The architecture below is known as a recurrent network:




A more popular version of this is the time delay neural network (TDNN):




These recurrent networks have been extremely important to allowing the integration of neural networks into the Markov model statistical framework we use in speech recognition. Such systems are known as **hybrid systems**.