# Neural Networks (CSC372)

## Unit 1: Introduction to Neural Network
## (4 Hrs.)

**Reference: Simon Haykin (3rd Edition)**

**By Kiran Bagale,**
**Department of IT, 2025**

# Overview

- Basics of Neural Networks
- Biological vs Artificial Neurons
- Models of a Neuron
- Neural Networks as Directed Graphs
- Feedback in Neural Networks
- Network Architectures
- Knowledge Representation
- Learning Processes & Learning Tasks

# Basics of Neural Networks

- What is a Neural Network?
  - A neural network is a massively parallel distributed processor.
  - It has a natural propensity for storing experiential knowledge.
  - Knowledge is acquired through learning; synaptic weights store this knowledge.
  - Inspired by human brain's structure and processing.
  - To achieve good performance, neural networks employ a massive interconnection of simple computing cells referred to as "neurons" or "processing units".
  - Learn from data, adapt to changes
  - Capable of pattern recognition, classification, etc.

## Thus definition of a neural network is viewed as an adaptive machine:

- *A neural network is a massively parallel distributed processor made up of simple processing units that has a natural propensity for storing experiential knowledge and making it available for use. It resembles the brain in two respects:*
    1. *Knowledge is acquired by the network from its environment through a learning process.*
    2. *Interneuron connection strengths, known as synaptic weights, are used to store the acquired knowledge.*

# Benefits of Neural Networks

**Key Properties and Capabilities**

1. **Nonlinearity**
2. **Input–Output Mapping**
3. **Adaptivity**
4. **Evidential Response**
5. **Contextual Information**
6. **Fault Tolerance**
7. **VLSI Implementability**
8. **Uniformity of Analysis and Design**
9. **Neurobiological Analogy**

# Biological Motivation

- Brain processes information differently from digital computers.
- Human brain can perform complex tasks quickly (e.g., vision in 100 - 200 ms).
- Example: Bat sonar and target detection via neural computations.

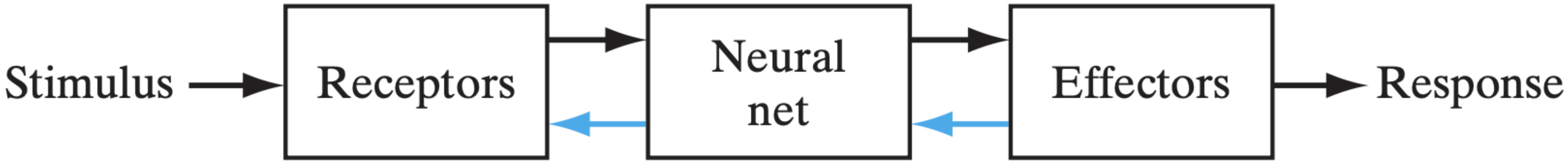



**FIGURE 1: Block diagram representation of nervous system.**

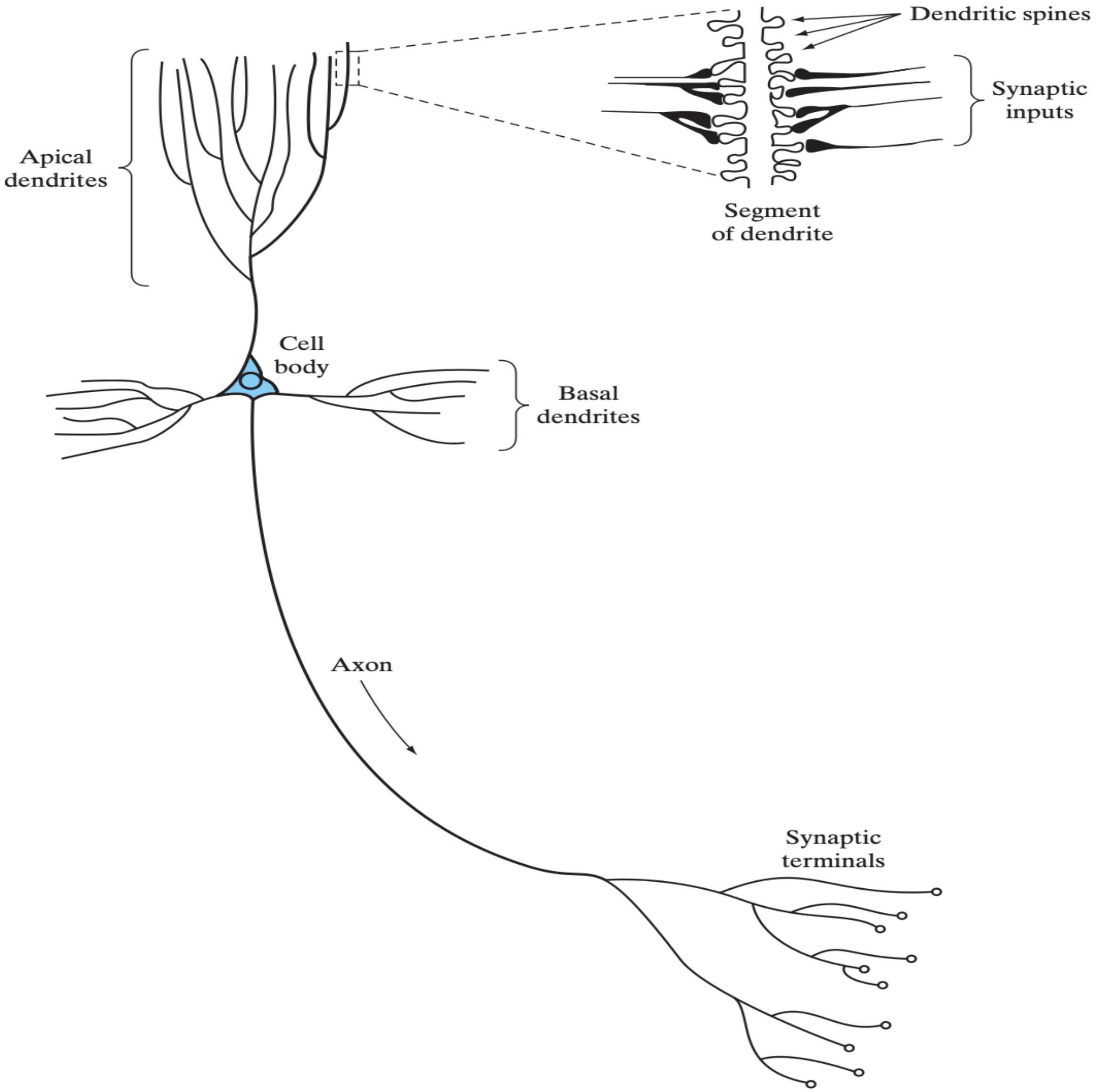



FIGURE 2: The pyramidal cell.

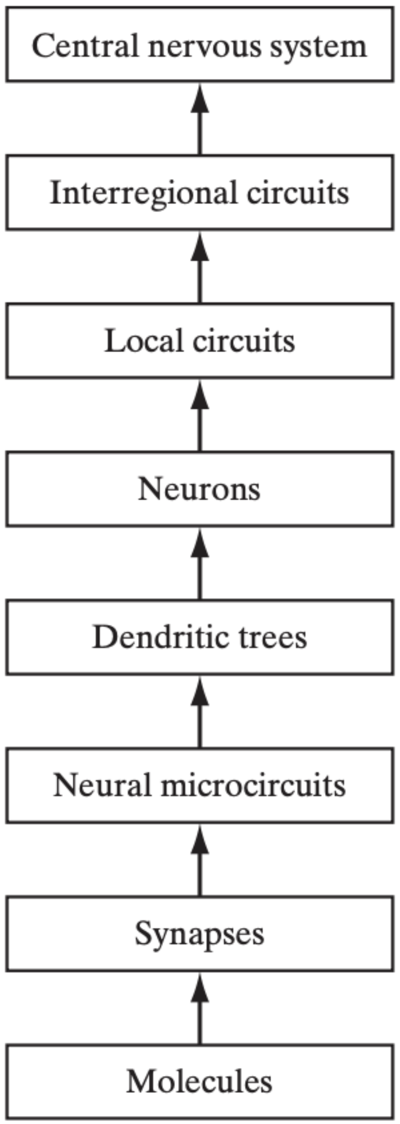



FIGURE 3: Structural organization of levels in the brain.

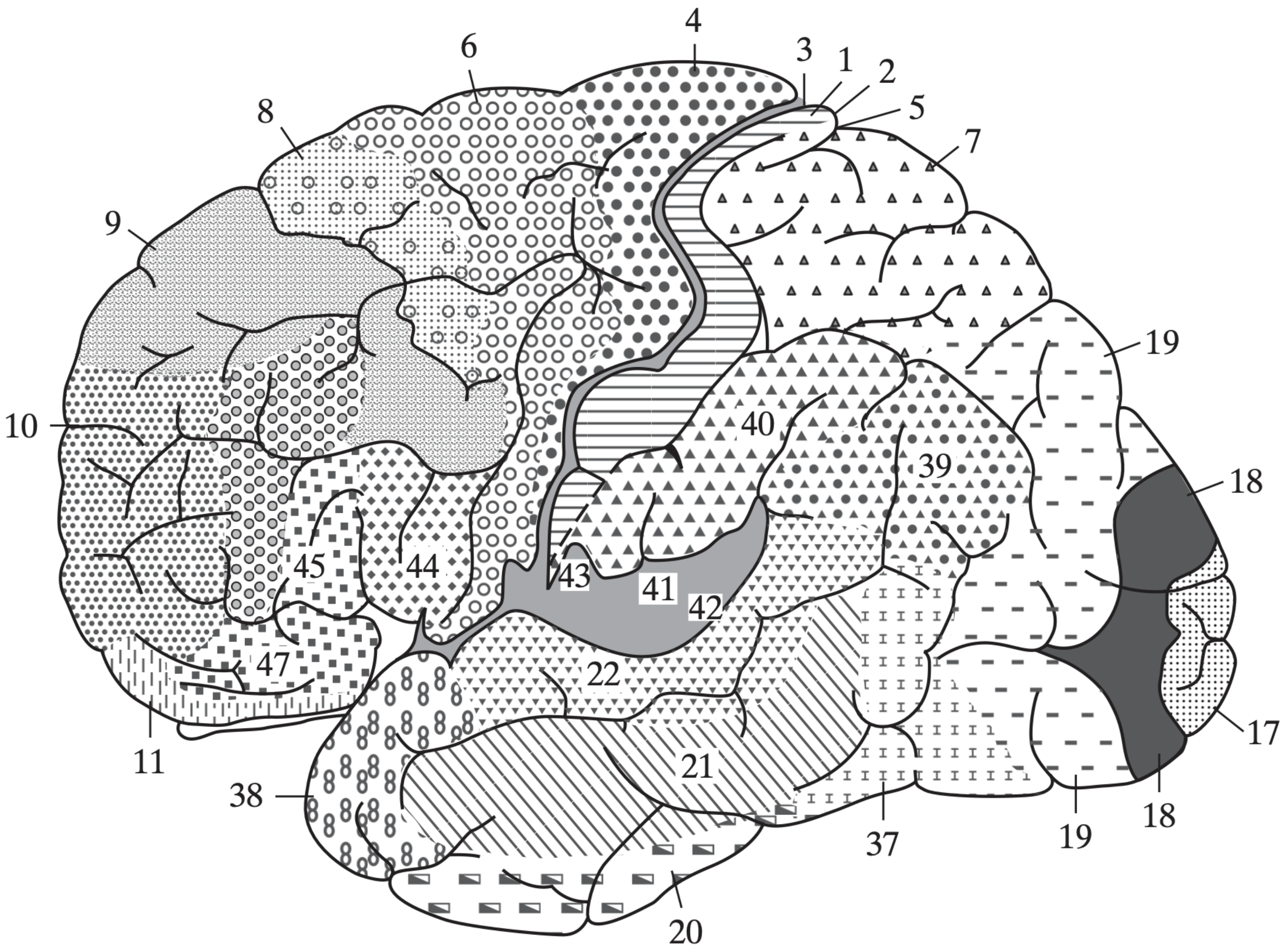



FIGURE 4: Cytoarchitectural map of the cerebral cortex. The different areas are identified by the thickness of their layers and types of cells within them. Some of the key sensory areas are as follows: Motor cortex: motor strip, area 4; premotor area, area 6; frontal eye fields, area 8. Somatosensory cortex: areas 3, 1, and 2. Visual cortex: areas 17, 18, and 19. Auditory cortex: areas 41 and 42. *(From A. Brodal, 1981; with permission of Oxford University Press.)*

# Human Brain vs Neural Networks

- Human Brain: ~100 billion neurons
- Artificial Neural Networks: Simplified abstraction
- Brain processes in parallel
- Synapse vs Weight

# Core Contrasts in Structure and Learning

| | | |
|---|---|---|
| **Structure** | Biological neurons with dendrites and axons | Mathematical models of neurons connected in layers |
| **Processing** | Parallel, distributed, and adaptive | Parallel (in hardware), typically layer-by-layer in software |
| **Learning Mechanism** | Synaptic plasticity (e.g., Hebbian learning) | Weight updates via algorithms (e.g., backpropagation) |
| **Speed** | Slow signal transmission (~m/s) | Fast computation (esp. with GPUs/TPUs) |
| **Energy Efficiency** | Highly efficient (~20W) | Energy-intensive training (especially deep networks) |
| **Adaptability** | Learns from few examples, strong generalization | Needs large datasets, often struggles with overfitting |
| **Fault Tolerance** | High (graceful degradation) | Moderate (improves with redundancy and dropout) |

The human brain excels in parallel, adaptive processing and efficient learning from limited examples. ANNs, while fast in computation, demand extensive data and energy. Brains gracefully degrade, unlike ANNs with moderate fault tolerance.

# Advanced Capabilities: Scalability and Creativity

**Human Brain**

- Naturally evolves with age and experience.
- Intuitive, context-rich understanding.
- Multi-modal, transfers learning easily across domains.
- Organic biological tissue platform.
- Capable of abstraction, imagination, and creativity.

**Artificial Neural Networks (ANNs)**

- Limited by compute power, data, and architecture.
- Limited context; must be explicitly modeled.
- Usually task-specific.
- Digital processors and specialized chips platform.
- Limited creativity; mimics patterns in training data.

The brain showcases natural evolution, rich context awareness, and inherent creativity. ANNs are constrained by hardware, require explicit context modeling, and mostly mimic training patterns. This highlights the unique flexibility of biological systems.

# Models of a Neuron

- A *neuron* is an information-processing unit that is fundamental to the operation of a neural network.
- Biological: Dendrites, Soma, Axon
- Artificial:
  - McCulloch-Pitts Neuron model: Simplified mathematical model.
  - Neuron consists of inputs, summation function, activation function, and output.
  - Inputs signals: $(x_1, x_2, ..., x_m)$,
  - Synaptic weights ($w_{k1}, w_{k2}, ..., w_{km}$)
  - $b_k$ is the bias
  - Weighted Sum: $u_k = \Sigma\,(w_{kj}x_j)$, Where j= 1,2,3,…,m
  - Output = $\varphi(w_{kj}x_j + b_k)$ where $\varphi$ is an activation function.

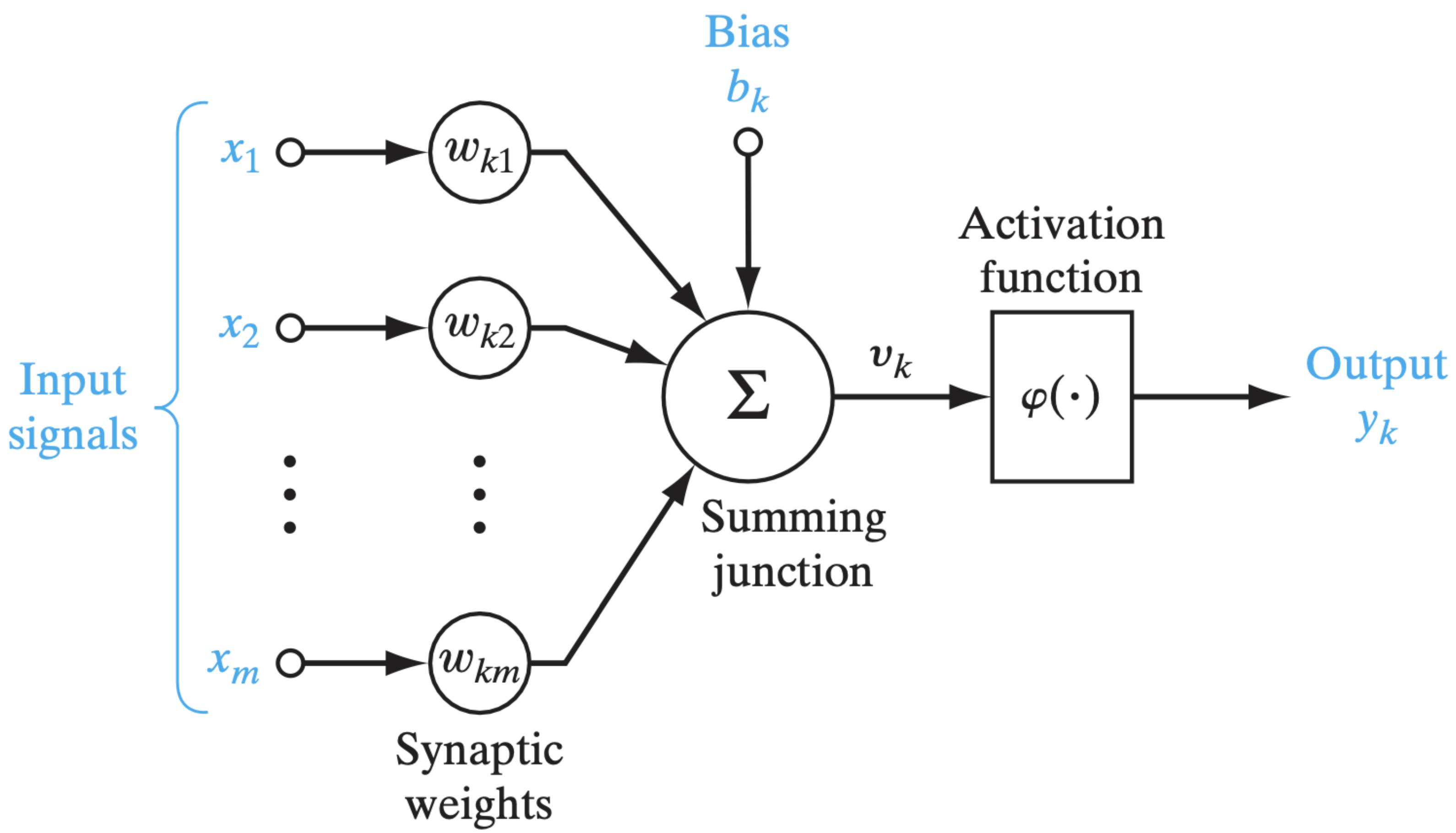



FIGURE 5: Nonlinear model of a neuron

- In mathematical terms,

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \ldots\ldots(1)$$

and

$$y_k = \varphi(u_k + b_k)\ldots\ldots(2)$$

Here, $b_k$ *is the bias and has* the effect of applying an *affine transformation* to the output $u_k$ of the linear combiner in the model of Fig. 5, as

$$v_k = u_k + b_k \ldots\ldots(3)$$

For $u_k = 0$, $v_k = b_k$. Equivalently, we may formulate the combination of Eqs. (1) to (3) as follows:

$$v_k = \sum_{j=0}^{m} w_{kj} x_j \ldots\ldots(4)$$

and $y_k = \varphi(v_k)\ldots\ldots(5)$

# Bias

- Imagine you're baking a cake 🍰 and following a recipe (your inputs and weights). But sometimes, **just following the recipe isn't enough** - maybe you like things a little sweeter or spicier. So, you **add a little extra ingredient** -something personal that shifts the result.
- That "extra something" in a neuron is called the **bias**.
- **Bias: The Neuron’s Secret Ingredient**
- The **bias** is a special number that is added **after** all the inputs have been multiplied by their weights and summed up.
- It gives the neuron **more flexibility** to make decisions.

$$Output = \varphi(w_1 \cdot x_1 + w_2 \cdot x_2 + \ldots + w_j \cdot x_j + b)$$

In Eq. (4), we have added a new synapse.
Its input is,

$$x_0 = +1 \dots\dots (6)$$

And its weight is,

$$w_{k0} = bk \dots\dots (7)$$

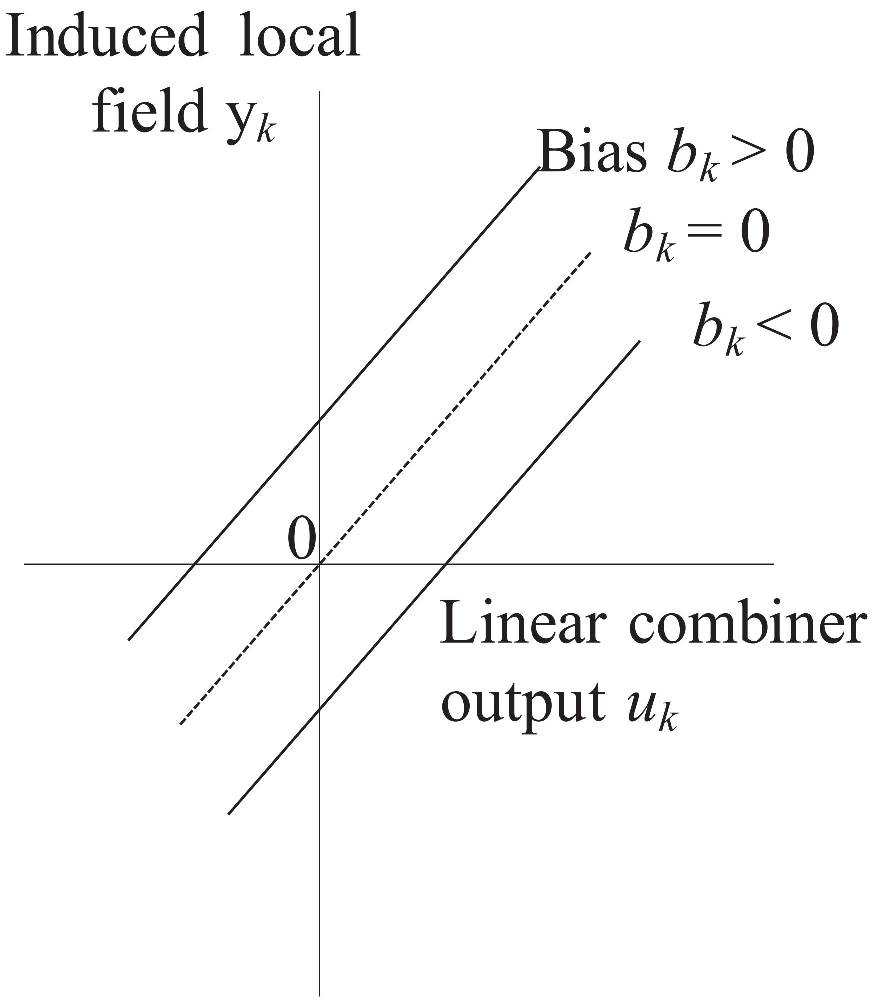



FIGURE 6: Affine transformation produced by the presence of a bias; note that $v_k = b_k$ at $u_k = 0$.

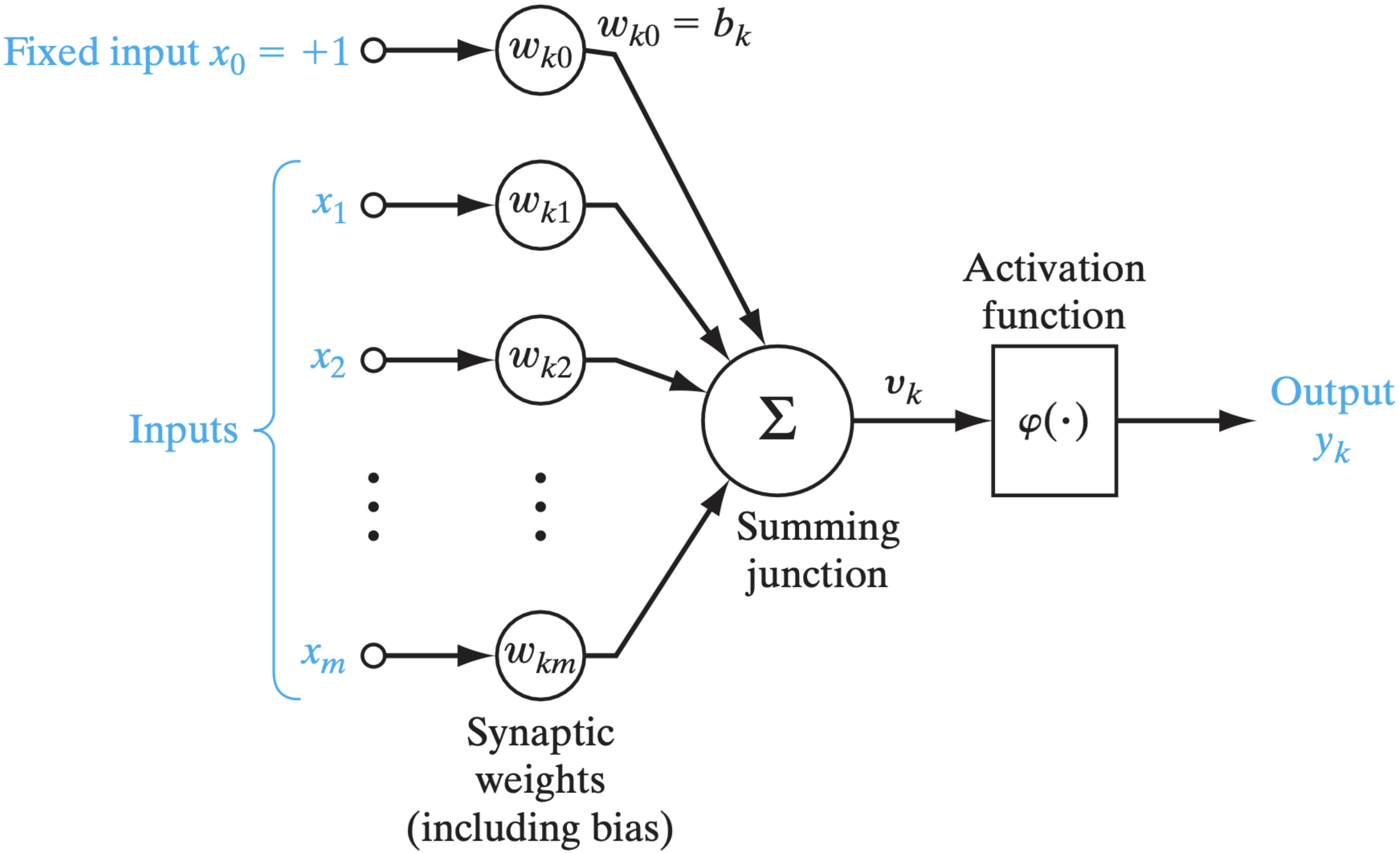



FIGURE 7:Another nonlinear model of a neuron; wk0 accounts for the bias bk.

# Types of Activation Function

- The activation function, denoted by (v), defines the output of a neuron in terms of the induced local field v.
- Two basic types of activation functions:

## 1. *Threshold Function*

$$\varphi(v) = \begin{cases} 1 \; if \; v \geq 0 \\ 0 \; if \; v < 0 \end{cases} \dots\dots (9)$$

This form of a threshold function is commonly referred to as a *Heaviside function.* Correspondingly, the output of neuron *k* employing such a threshold function is expressed as

$$y_k(v) = \begin{cases} 1 \; if \; vk \geq 0 \\ 0 \; if \; vk < 0 \end{cases} \dots\dots (10)$$

where $v_k$ is the induced local field of the neuron; that is, $v_k = \sum_{j=1}^{m} w_{kj} x_j + \mathrm{bk}$

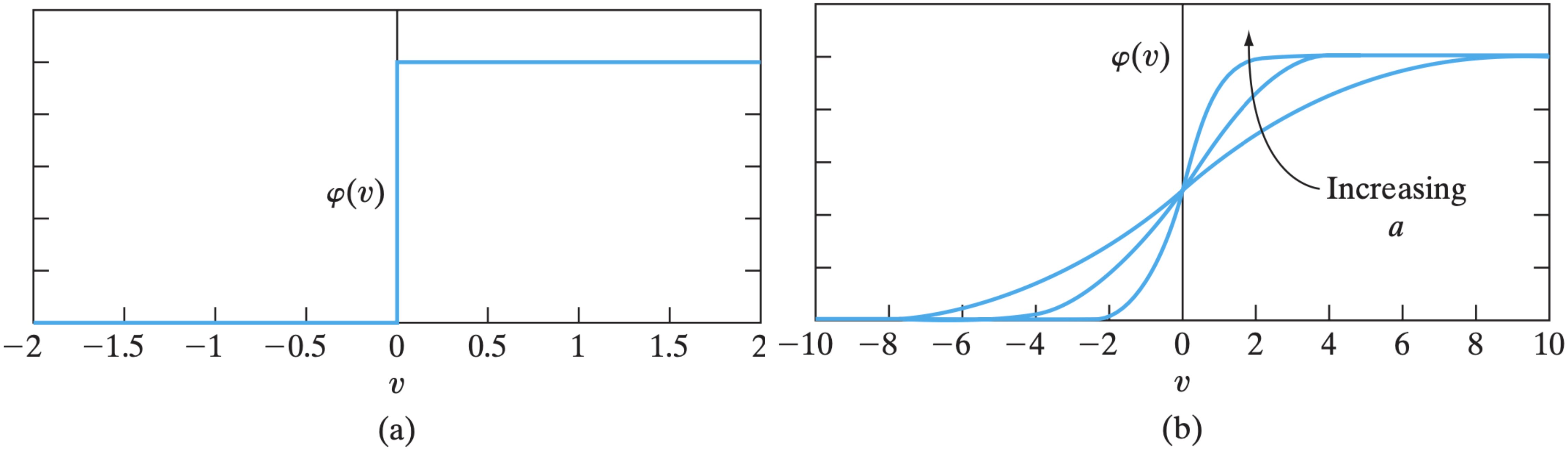



Figure 8: (a) Threshold function. (b) Sigmoid function for varying slope parameter $a$.

# 2. ***Sigmoid Function***

- Graph is “S”-shaped
- The most common form of activation function used in the construction of neural net- works.
- It is defined as a strictly increasing function that exhibits a graceful balance be- tween linear and nonlinear behavior.
- An example of the sigmoid function is the *logistic function*,5 defined by

$$\varphi(v) = \frac{1}{1 + \exp(-av)} \ldots\ldots (11)$$

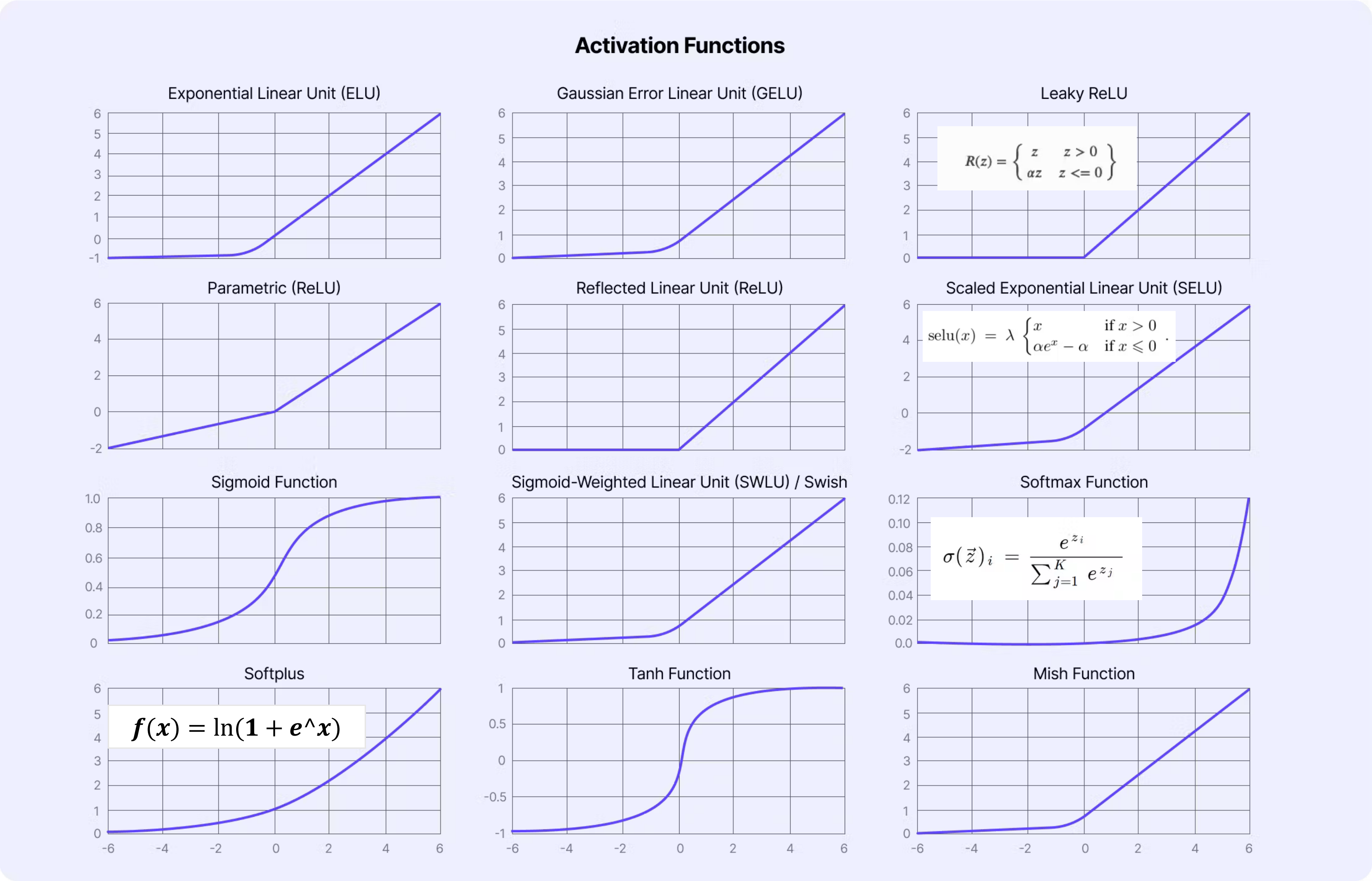
where *a* is the *slope parameter* of the sigmoid function.

$$\varphi(v) = \begin{cases} 1\ if\ v > 0 \\ 0\ if\ v = 0 \\ -1\ if\ v < 0 \end{cases} \ldots\ldots (12)$$

which is commonly referred to as the *signum function.*

For the corresponding form of a sigmoid function, we may use the *hyperbolic tangent function*, defined by

$$\varphi(v) = \tanh(v)$$


Activation Functions
Exponential Linear Unit (ELU)
Gaussian Error Linear Unit (GELU)
Leaky ReLU
$R(z) = \begin{Bmatrix} z & z > 0 \\ \alpha z & z <= 0 \end{Bmatrix}$
Parametric (ReLU)
Reflected Linear Unit (ReLU)
Scaled Exponential Linear Unit (SELU)
$\mathrm{selu}(x) = \lambda \begin{cases} x & \text{if } x > 0 \\ \alpha e^x - \alpha & \text{if } x \leqslant 0 \end{cases}.$
Sigmoid Function
Sigmoid-Weighted Linear Unit (SWLU) / Swish
Softmax Function
$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$
Softplus
$f(x) = \ln(1 + e^\wedge x)$
Tanh Function
Mish Function

# Neural Networks Viewed as Directed Graphs

- Signal-flow graphs, with a well-defined set of rules, were originally developed by Mason (1953, 1956) for linear networks.
- A *signal-flow graph* is a network of directed *links* (*branches*) that are interconnected at certain points called *nodes*.

i.e., Neurons represented as nodes, synaptic connections as directed edges.

  - Nodes = Neurons, Edges = Weighted connections

- Helps in visualizing flow of information in the network.
  - Directed flow of information
- Used to describe feedforward and feedback (recurrent) networks.
  - Layers: Input, Hidden, Output

# Neural Networks Viewed as Directed Graphs (Cont.…)

## Three basic rules:

***Rule 1.*** *A signal flows along a link only in the direction defined by the arrow on the link.*

- Two different types of links may be distinguished:
  - ***Synaptic links***, **whose behavior is governed by a *linear* input–output relation. Specifically, the node signal $x_j$ is multiplied by the synaptic weight $w_{kj}$ to produce the node signal $y_k$, as illustrated in Fig. 9a.**
  - ***Activation links***, **whose behavior is governed in general by a *nonlinear* input–output relation. This form of relationship is illustrated in Fig. 9b, where $\varphi(\cdot)$ is the non- linear activation function.**

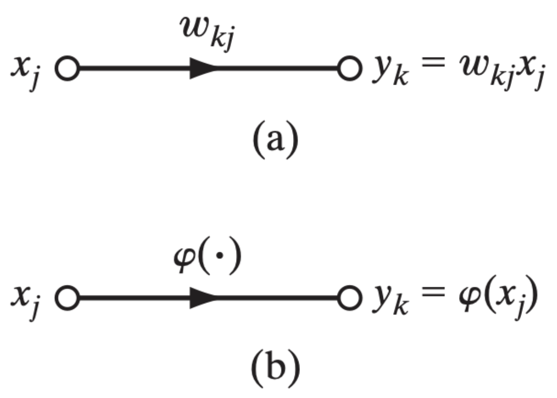
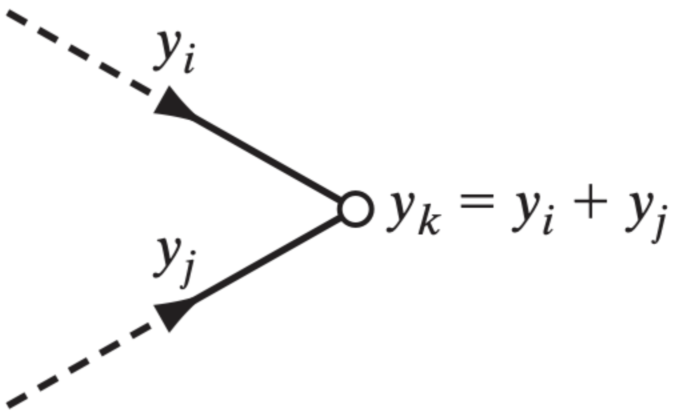


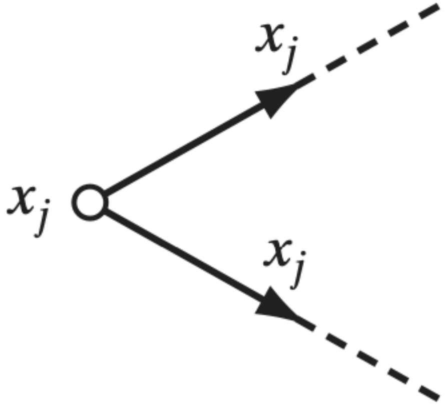

FIGURE 9: illustrating basic rules for the construction of signal-flow graphs.

***Rule 2.*** *A node signal equals the algebraic sum of all signals entering the pertinent node via the incoming links.*




(c)

***Rule 3.*** *The signal at a node is transmitted to each outgoing link originating from that node, with the transmission being entirely independent of the transfer functions of the outgoing links.*




(d)

*A neural network is a directed graph consisting of nodes with interconnecting synaptic and activation links and is characterized by four properties:*

1. *Each neuron is represented by a set of linear synaptic links, an externally applied bias, and a possibly nonlinear activation link. The bias is represented by a synaptic link connected to an input fixed at* 1.
2. *The synaptic links of a neuron weight their respective input signals.*
3. *The weighted sum of the input signals defines the induced local field of the neuron in question.*
4. *The activation link squashes the induced local field of the neuron to produce an output.*

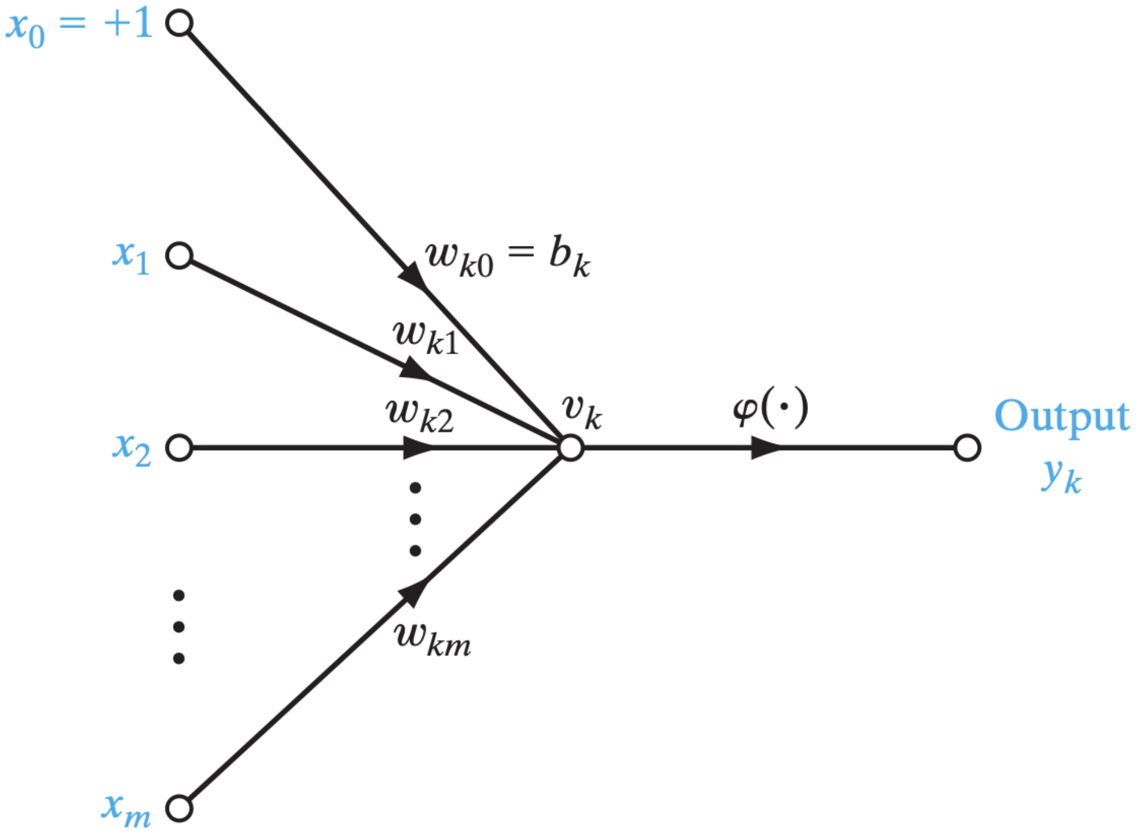



**FIGURE 10: Signal-flow graph of a neuron.**

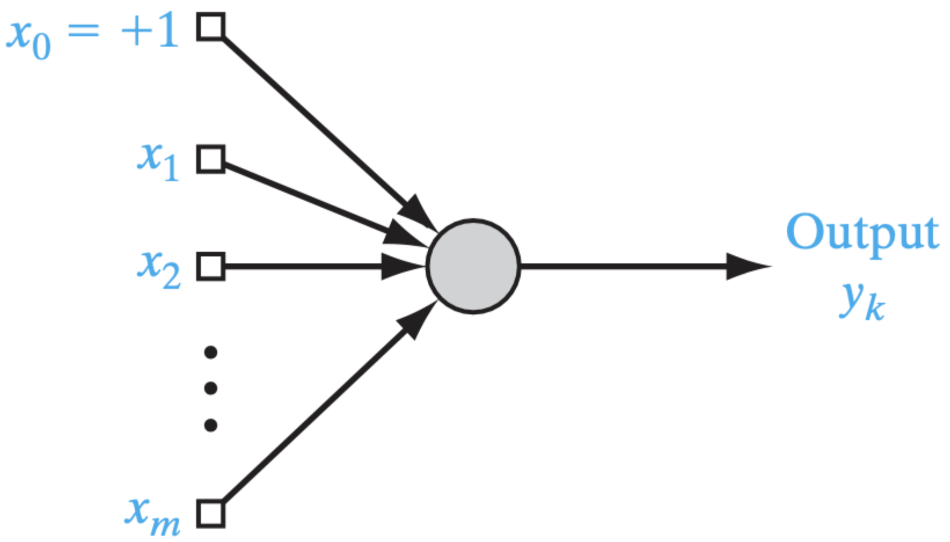



**FIGURE 11: Architectural graph of a neuron.**

# Feedback in Neural Networks

- Feedback introduces memory into the network (recurrent architecture).
- Indeed, feedback occurs in almost every part of the nervous system of every animal (Freeman, 1975).
- Network types: Feedforward, Feedback (Recurrent), Lateral connections.
- Architecture defines learning capacity and performance.

Feedforward: No cycles (MLP)

- Feedback: Cycles present (e.g., Hopfield)

- Used for memory and sequences

- Suppose we have a linear system consisting of a forward path and a feedback path that are characterized by the “operators” **A** and **B**, respectively.
- In particular, the output of the forward channel determines in part its own output through the feedback channel.

$$y_k(n) = \mathbf{A}[x_j'(n)] \tag{16}$$

$$x_j'(n) = x_j(n) + \mathbf{B}[y_k(n)] \tag{17}$$

where the square brackets are included to emphasize that **A** and **B** act as *operators*.
Eliminating $x_j'(n)$ between Eqs. (16) and (17), we get

$$y_k(n) = \frac{\mathbf{A}}{1 - \mathbf{AB}}[x_j(n)] \tag{18}$$

Consider, for example, the single-loop feedback system having **A** as a fixed weight $w$ and **B** as a *unit-delay operator* $z$-1, whose output is delayed with respect to the input by one time unit.

$$\frac{\mathbf{A}}{1-\mathbf{AB}} = \frac{w}{1-wz^{-1}}$$
$$= w(1-wz^{-1})^{-1}$$

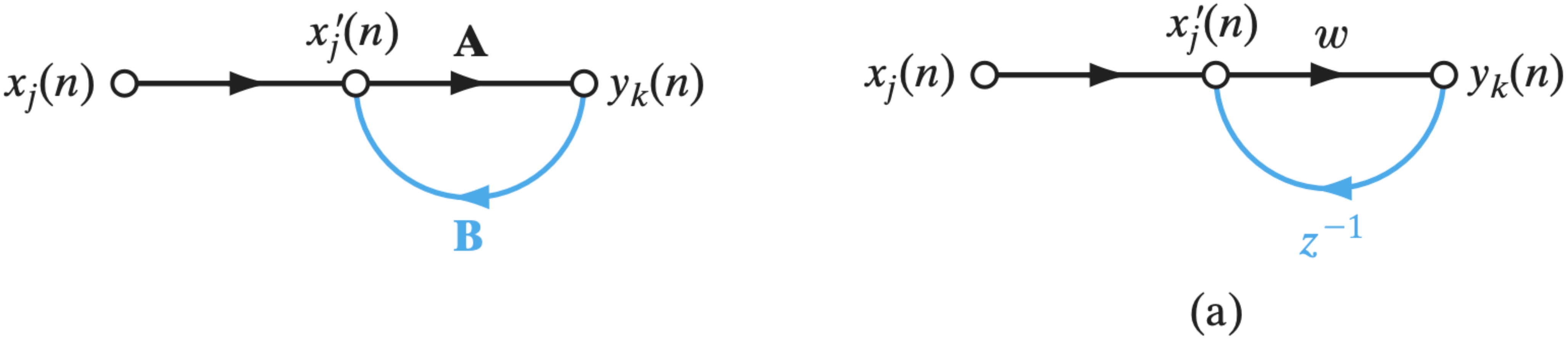



**Figure 12: Signal-flow graph of a single-loop feedback system.**

Using the binomial expansion for $(1 \; - wz^{-1})^{-1}$, we may rewrite the closed-loop operator of the system as

$$\frac{A}{1 - AB} = w \sum_{l=0}^{\infty} w^l z^{-1} \tag{19}$$

substituting Eq. (19) into (18), we get,

$$y_k(n) = w \sum_{l=0}^{\infty} w^l z^{-l}[x_j(n)] \tag{20}$$

From the definition of $z^{-1}$, we have

$$z^{-l}[x_j(n)] = x_j(n - l) \tag{21}$$

Using (21) in (20), we get,

$$y_k(n) = \sum_{l=0}^{\infty} w^{l+1} x_j(n - l) \tag{22}$$

# Network Architectures

- ✓Single-Layer Feedforward Networks
- ✓Multilayer Feedforward Networks
- ✓Recurrent Networks (RNNs)
- ✓Convolution Networks (CNNs)

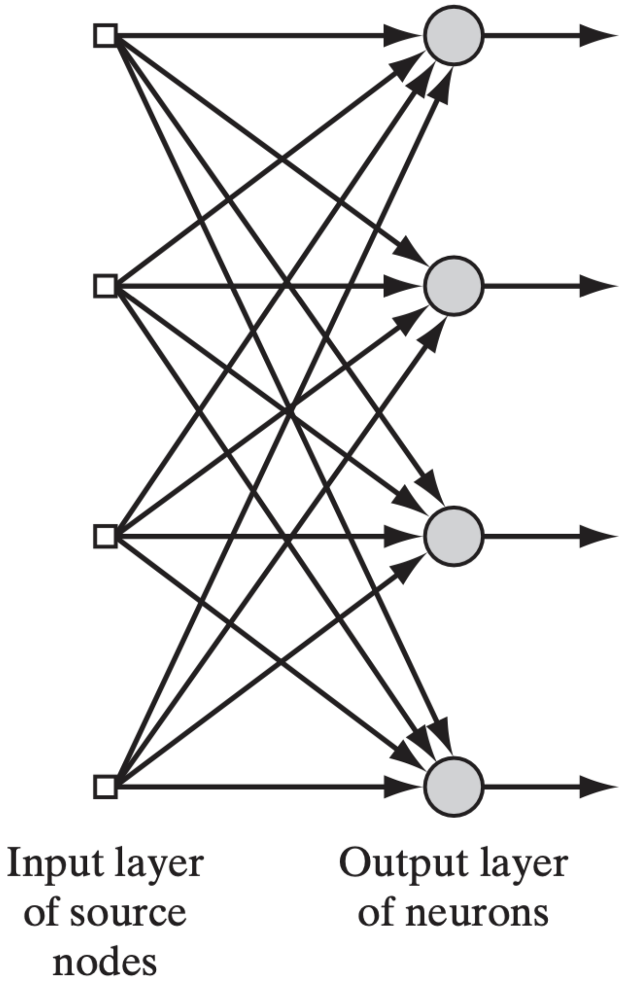



**FIGURE 15: Feedforward network with a single layer of neurons.**

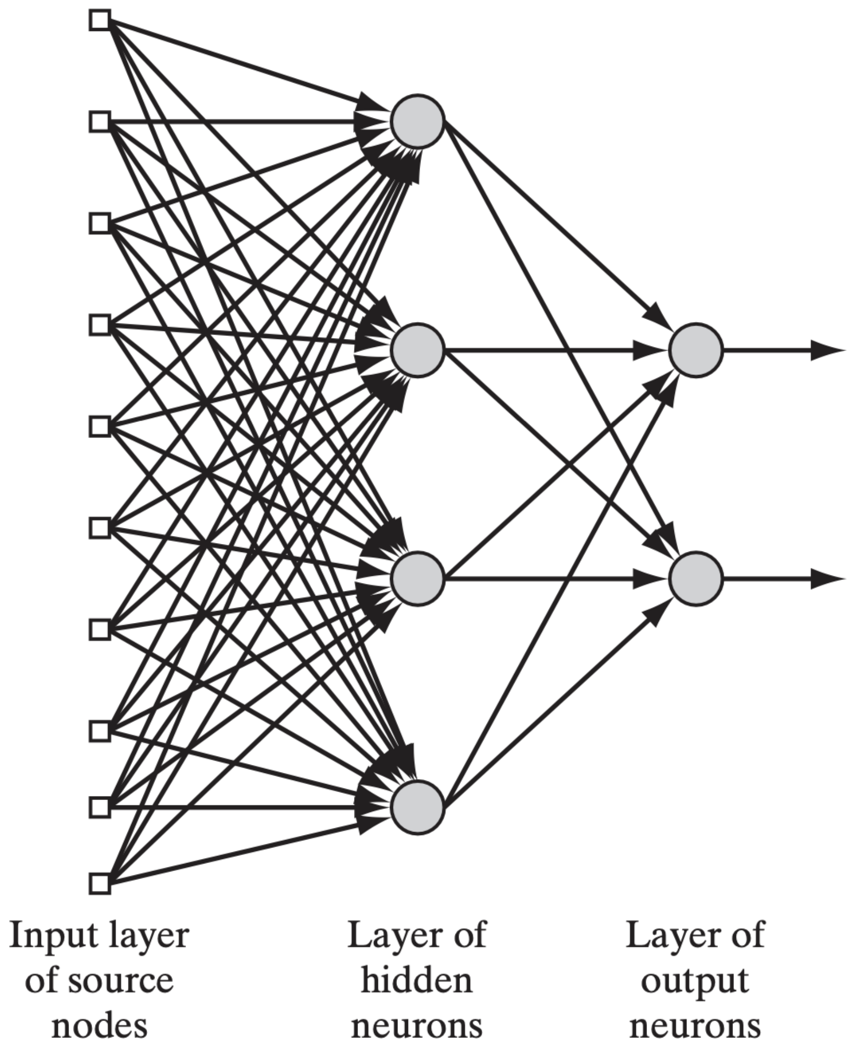



**FIGURE 16: Fully connected feedforward network with one hidden layer and one output layer.**

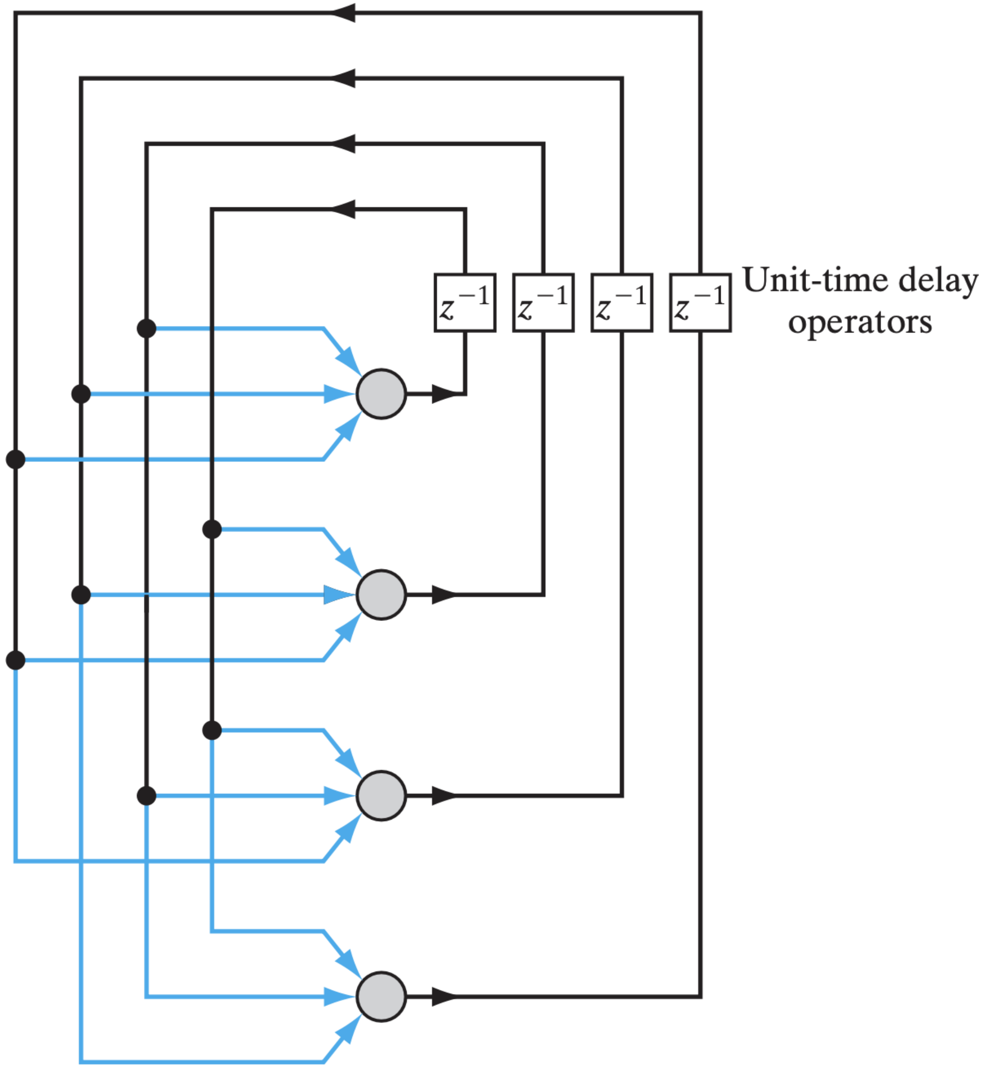



**FIGURE 17: Recurrent network with no self-feedback loops and no hidden neurons.**

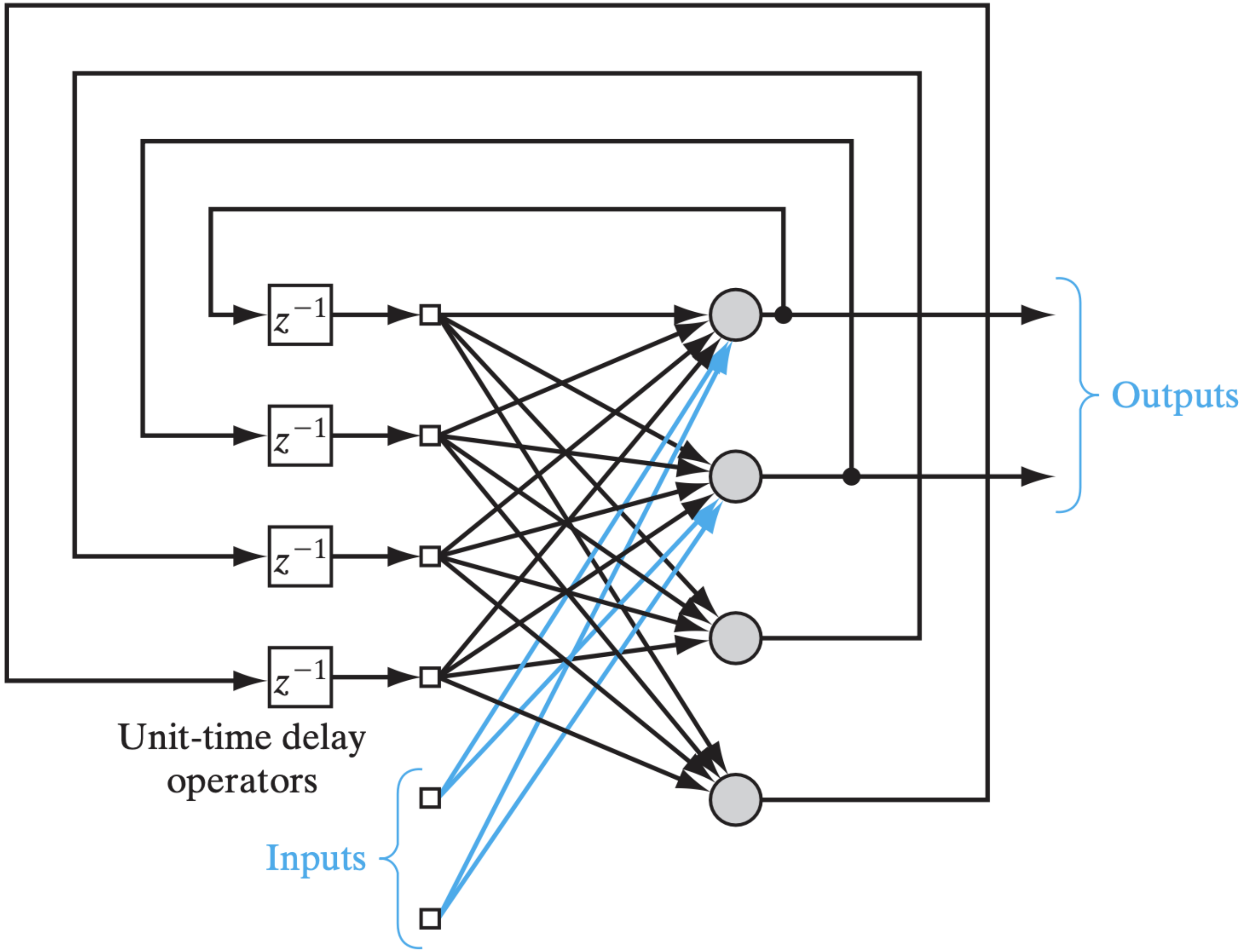



**FIGURE 18: Recurrent network with hidden neurons.**

# Knowledge Representation

*Knowledge refers to stored information or models used by a person or machine to interpret, predict, and appropriately respond to the outside world.*

- ✓ Stored in connection weights
- ✓ Adjusted during training
- ✓ Represents input-output mappings
- ✓Knowledge is stored in synaptic weights.
- ✓ Can be local (one neuron) or distributed (across network).
- ✓ Generalization from learned data is key.

# Roles of Knowledge Representation

- **Rule 1.** Similar inputs (i.e., patterns drawn) from similar classes should usually produce similar representations inside the network, and should therefore be classified as belonging to the same class.
- **Rule 2.** Items to be categorized as separate classes should be given widely different representations in the network.
- **Rule 3.** If a particular feature is important, then there should be a large number of neurons involved in the representation of that item in the network.
- **Rule 4.** Prior information and invariances should be built into the design of a neural network whenever they are available, so as to simplify the network design by its not having to learn them.

# Learning Processes

✓**Learning with a Teacher -** Supervised Learning

✓**Learning without a Teacher**

1. Reinforcement Learning
2. Unsupervised Learning

- Supervised Learning: Learning with a teacher (input-output pairs).
- Reinforcement Learning: Learning via feedback from environment.
- Unsupervised Learning: Learning patterns from input only.

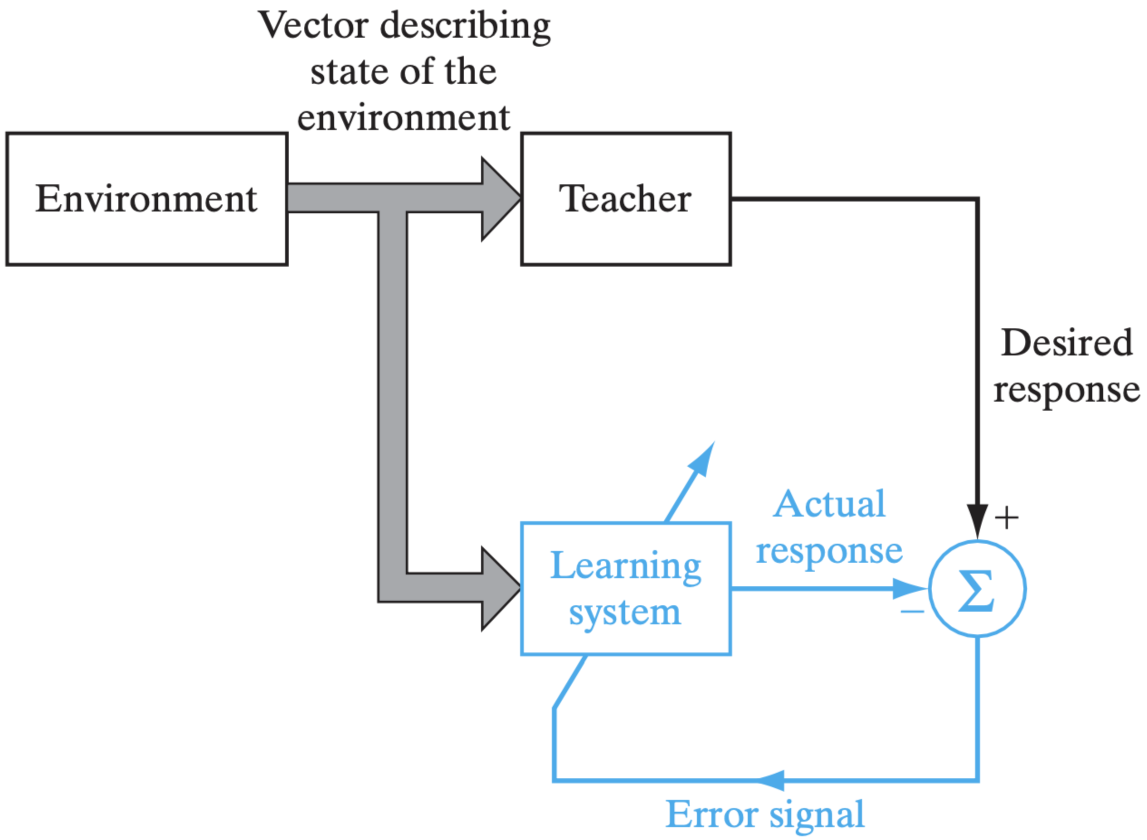



Fig: Block diagram of learning with a teacher

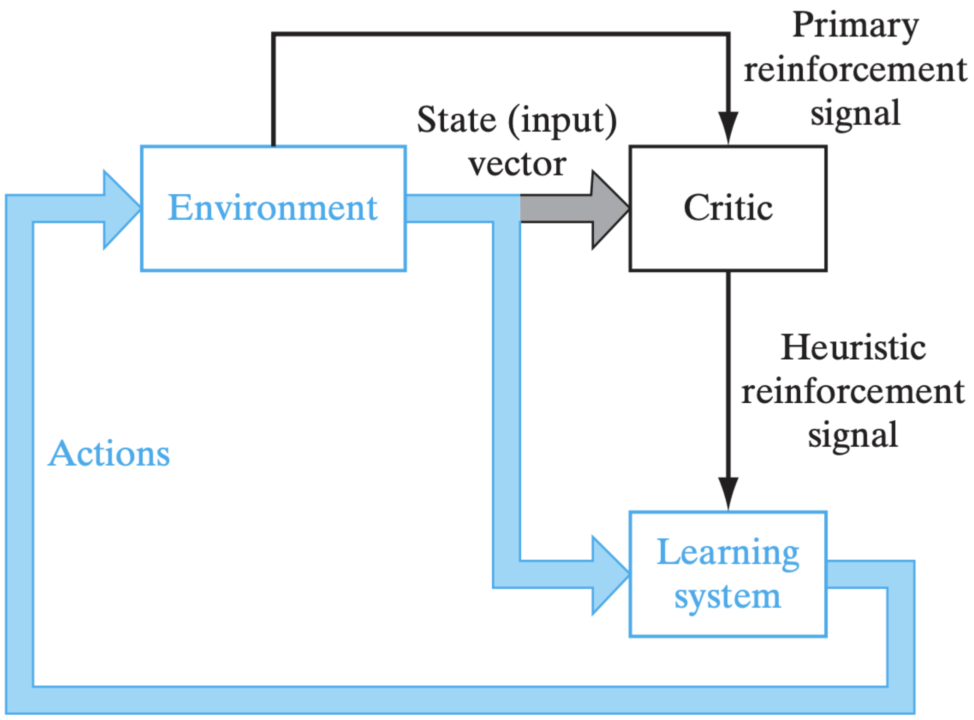



**Fig: Block diagram of reinforcement learning; the learning system and the environment are both inside the feedback loop.**

# Learning Tasks

✓**Pattern Association:** *auto-association* and *hetero-association*

✓**Pattern Recognition**

✓**Function Approximation:** *System identification and Inverse Modeling*

✓**Clustering**

✓**Control Tasks:** *Indirect and Direct learning*

✓**Beamforming**

- Pattern Association, An *associative memory* is a brainlike distributed memory that learns by *association.*
- Pattern Recognition, clustering, function approximation, prediction.
- Different tasks require different network types and learning methods.
- Adaptivity is crucial in non-stationary environments.
- Beamforming is used to distinguish between the spatial properties of a target signal and background noise.

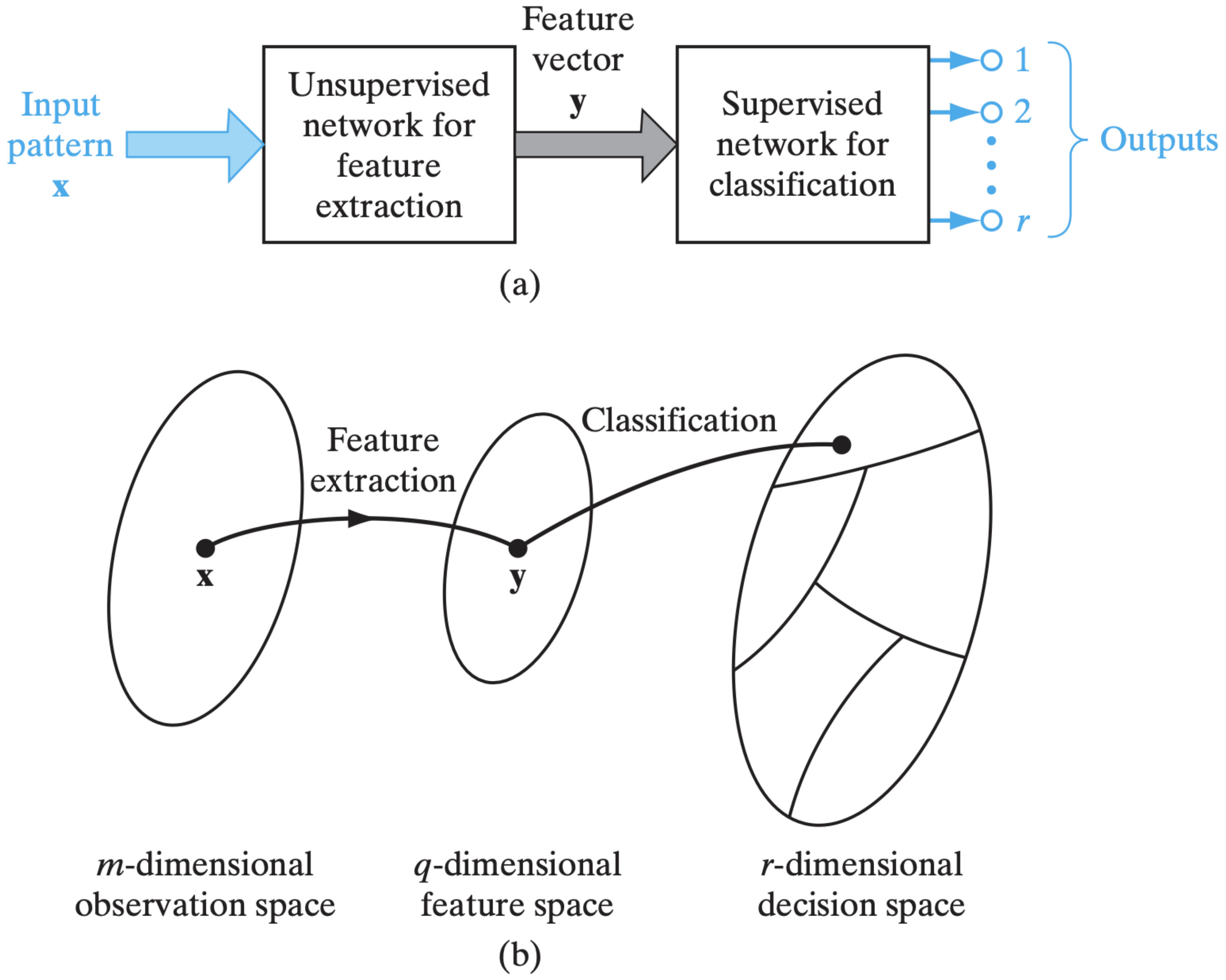



**Fig: Illustration of the classical approach to pattern classification.**

# Reading & Homework

- Read Chapter 1 (Haykin)
- Assignment: Compare biological and artificial neurons
- Quiz: Learning processes

# Thank You!!!