EENG582
Artificial Neural Networks

Introduction

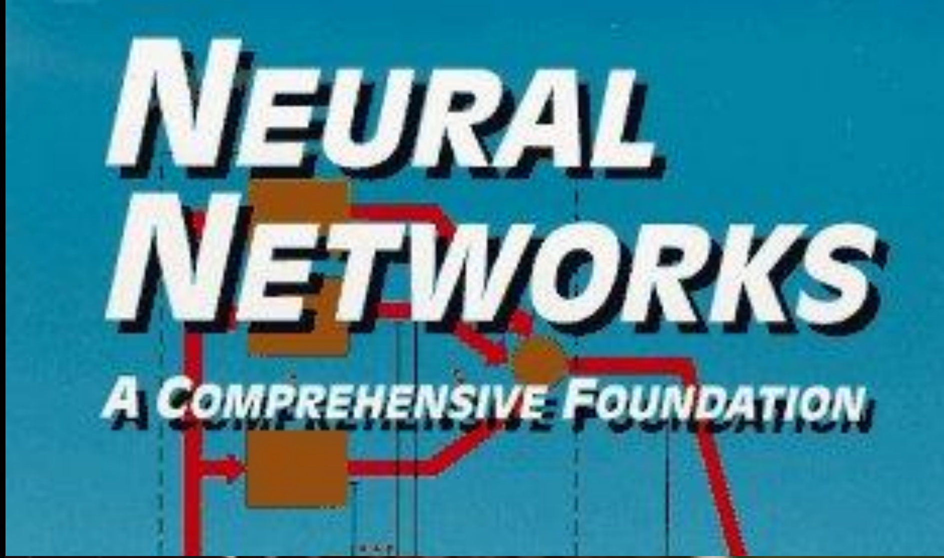
Sections 1.7 – 1.9 – Models of Neurons

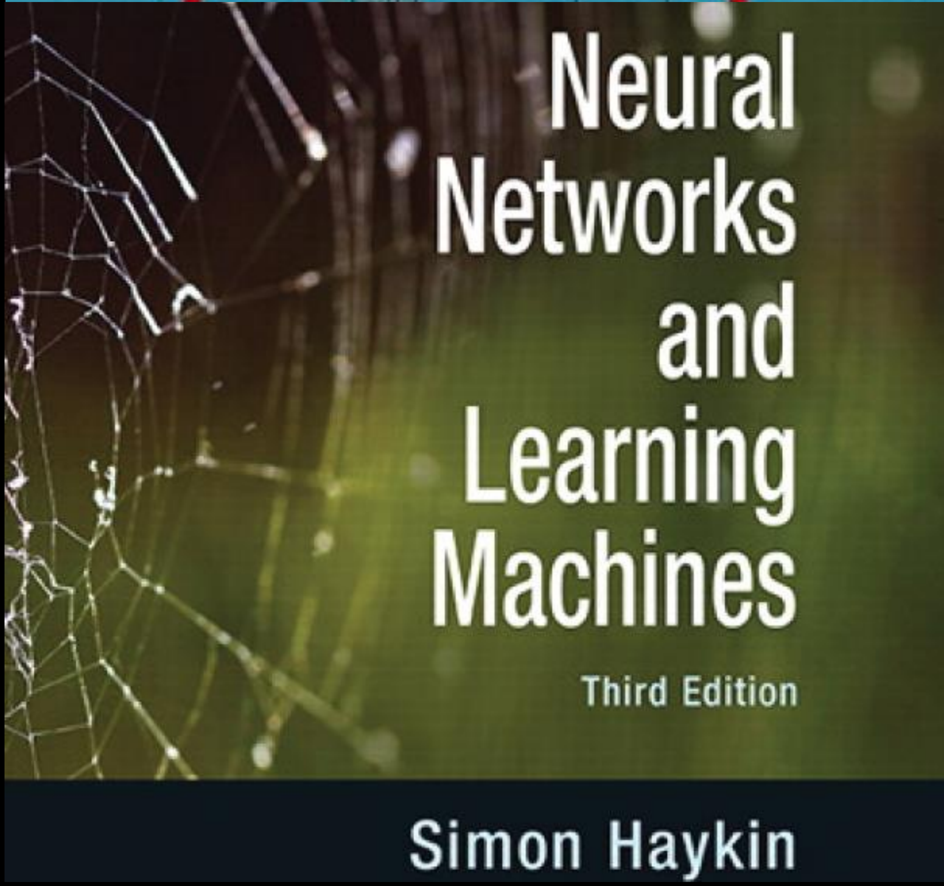
Artificial Neural Networks

Prof. Dr. Hasan AMCA

Electrical and Electronic Engineering Department
(ee.emu.edu.tr)

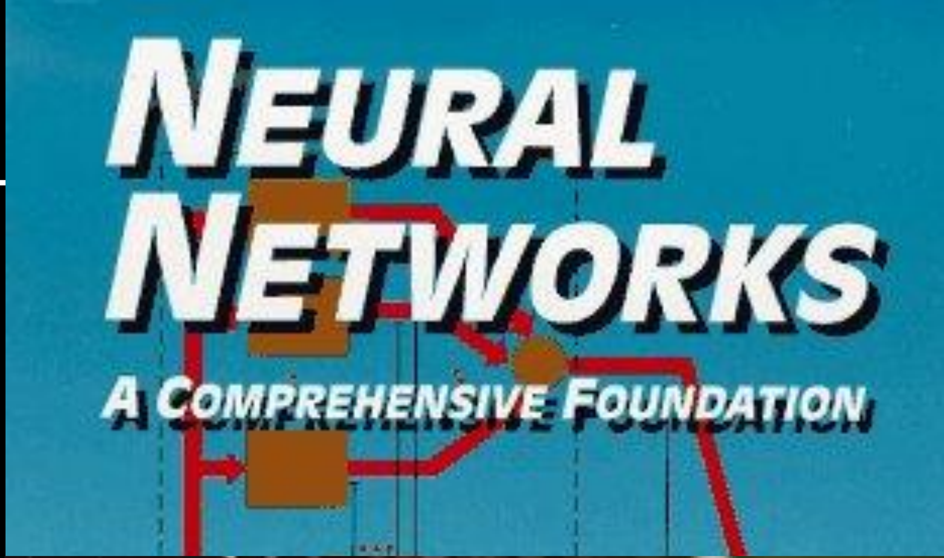
Eastern Mediterranean University
(emu.edu.tr)

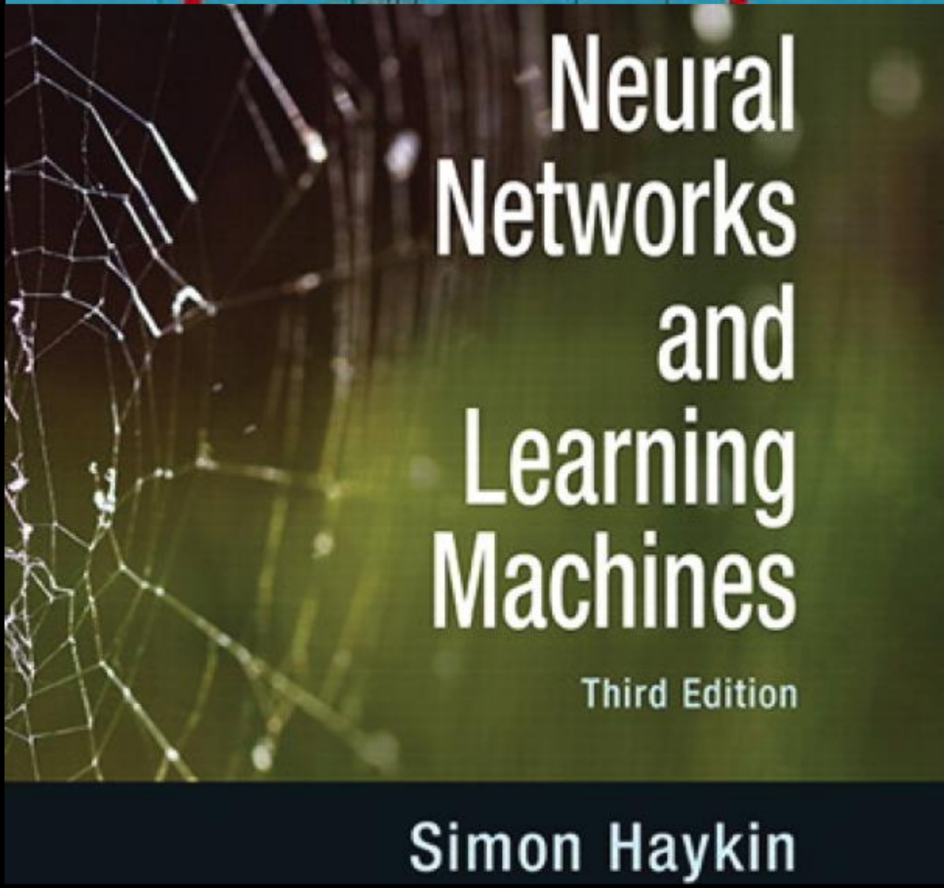




1 Introduction

1.1 What Is a Neural Network?
1.2 Human Brain
1.3 Models of a Neuron
1.4 Neural Networks Viewed as Directed Graphs
1.5 Feedback
1.6 Network Architectures
1.7 Knowledge Representation
1.8 Artificial Intelligence and Neural Networks
1.9 Historical Notes
Notes and References
Problems





1.7 Knowledge Representation

- The generic definition of "knowledge«

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

- The primary characteristics of knowledge representation are twofold
  1) what information is actually made explicit; and
  2) how the information is physically encoded for subsequent use
- In real-world applications of "intelligent" machines, it can be said that a good solution depends on a good representation of knowledge

3. Recurrent Networks

- A major task for a neural network is to learn a model of the world (environment) in which it is embedded and to maintain the model sufficiently consistent with the real world so as to achieve the specified goals of the application of interest

- Knowledge of the world consists of two kinds of information:

1. The known world state, represented by facts about what is and what has been known (the prior information)

2. Observations (measurements) of the world, obtained by means of sensors designed to probe the obsersation environment (subject to sensor noise, errors and system imperfections)

Training Data or Training Sample

- A set of input-output pairs, with each pair consisting of an input signal and the corresponding desired response, is referred to as a set of training data or training sample
- To illustrate how such a data set can be used, consider, for example, the handwritten digit recognition problem where the input signal is an image with black or white pixels, representing one of 10 digits. The desired response is the "identity" of the particular digit whose image is presented to the network as the input signal
- In this example, design of a neural network may proceed as follows:

Design of a Neural Network for Character Recognition

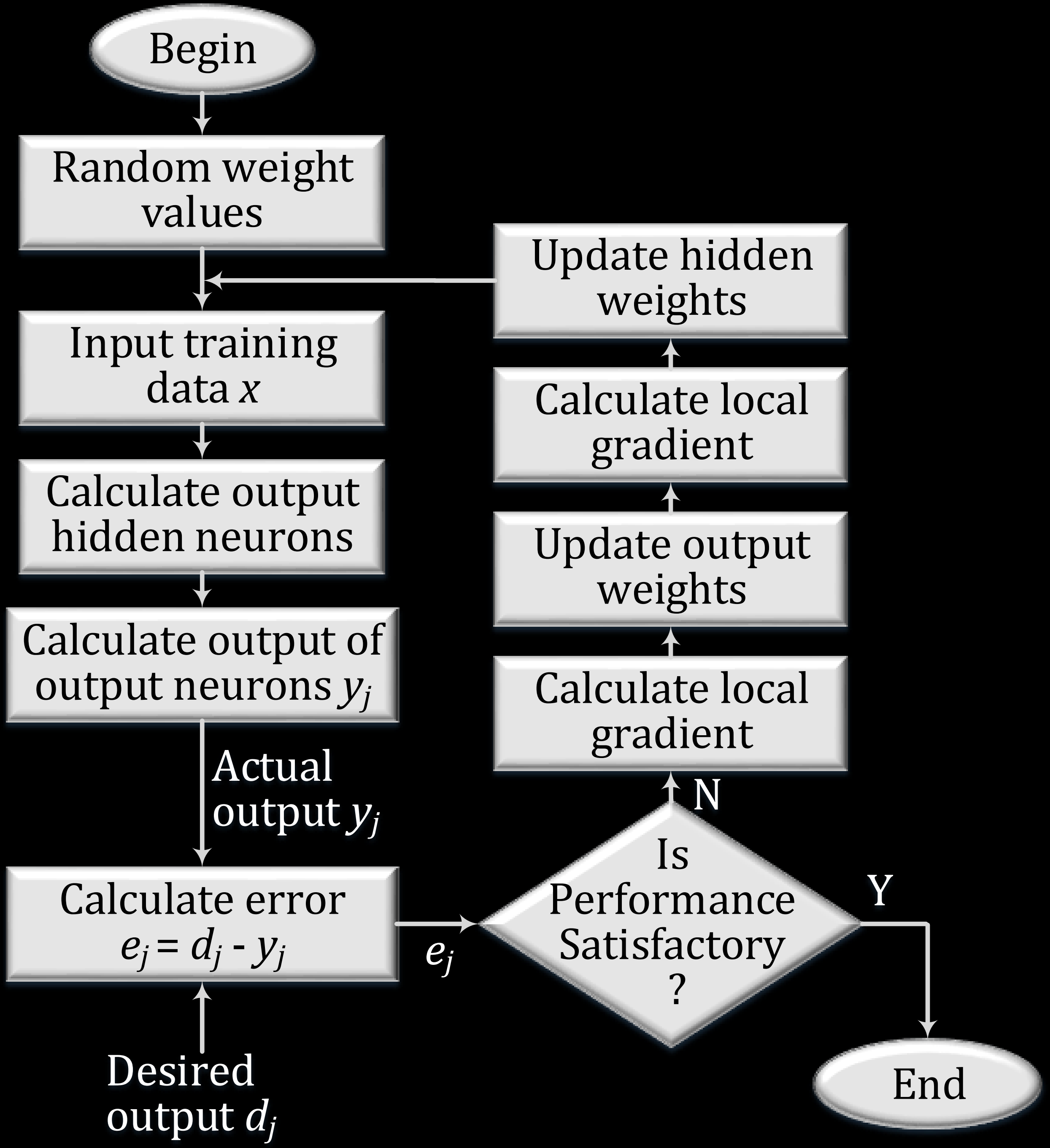
1. **Select the appropriate architecture** for the neural network,
   - ~ with an input layer consisting of source nodes equal in number to the pixels of an input image,
   - ~ and an output layer consisting of 10 neurons (one for each digit),
   - ~ A subset of examples is then used to train the network by means of a suitable algorithm.
   - ~ This phase of the network design is called **learning**

2. **Test the recognition performance** of the trained network with data not seen before
   - ~ Specifically, an input image is presented to the network, but this time it is not told the identity of the digit to which that particular image belongs.
   - ~ Performance of the network is then assessed by comparing the digit recognition reported by the network with actual identity of the digit
   - ~ This second phase of the network operation is called **generalization**

Design of a Neural Network for Character Recognition

1. **Select the appropriate architecture** for the NN
   - ~ input layer consisting of #nodes = #pixels of input image
   - ~ an output layer consisting of 10 neurons (1 for each digit),
   - ~ Use examples to train the network (**learning)**

2. **Test the recognition performance** with data not seen before
   - ~ Unknown input images are used to identified the digit.
   - ~ Network performance is assessed by comparing the identified digit with the actual digit.
   - ~ This second phase is called **generalization**



Rules for Knowledge Representation – Rule 1

Rule 1: Knowledge representation inside an artificial neural network

- Similar inputs from similar classes should produce similar representations inside the network and should therefore be classified as belonging to the same category.
- A common measure of similarity is the Euclidian distance
- To be specific, let $\mathbf{x}_i$ denote an m-by-1 vector

$$\mathbf{x}_i = [x_{i1}, x_{i2}, \dots, x_{im}]^T$$

- Where all elements are real. The superscript T denotes matrix transposition
- Vector $\mathbf{x}_\mathrm{i}$ defines a point in an m-dimensional space called Euclidean space
- Euclidean distance between a pair of m-by-1 vectors $\mathbf{x}_\mathrm{i}$ and $\mathbf{x}_\mathrm{j}$ is defined by

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\| = \left[\sum_{k=1}^{m} \left(x_{ik} - x_{jk}\right)^2\right]^{1/2} \qquad (1.23)$$

Where x_{ik} and x_{jk} are the kth element of the input vectors $\mathbf{x}_i$ and $\mathbf{x}_j$.

Measure of Similarity Between Patterns

- The similarity between the inputs represented by the vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ is defined as the reciprocal of the Euclidean distance $d(\mathbf{x}_i, \mathbf{x}_j)$. The closer the individual

- Rule 1 states that if the vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ are similar, they should be assigned to the same category (class).

- Another measure of similarity is based on the *dot product* or *inner product*.

- Given a pair of vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ of same dimension, their inner product is

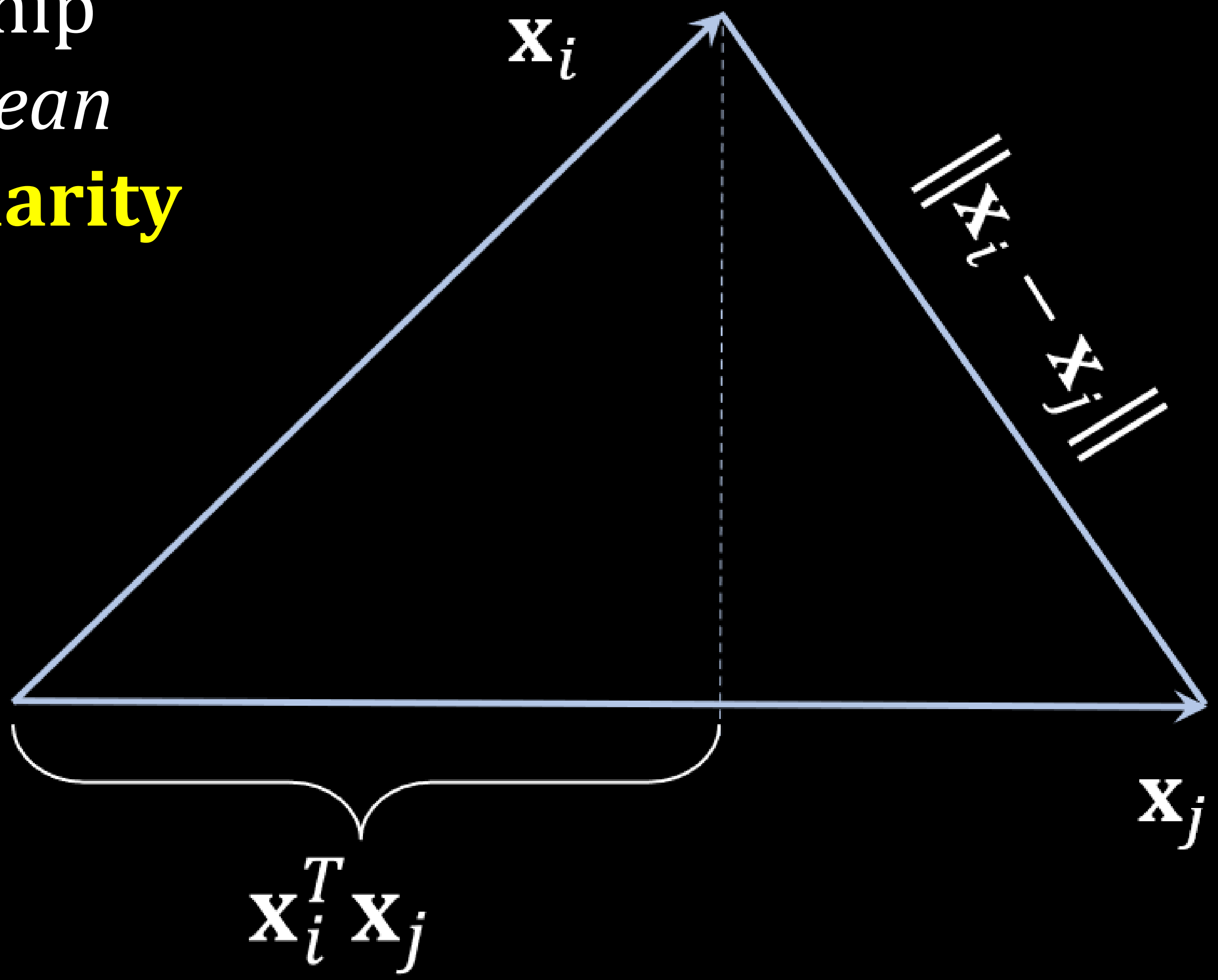
$$(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j = \sum_{k=1}^{m} x_{ik} x_{jk} \qquad (1.24)$$

The inner product $(\mathbf{x}_i, \mathbf{x}_j)$ divided by $\|\mathbf{x}_i\|\|\mathbf{x}_j\|$ is the cosine of the angle subtended between the vectors $(\mathbf{x}_i, \mathbf{x}_j)$.

See Fig. 1.19 for illustration.

Measure of Similarity Between Patterns

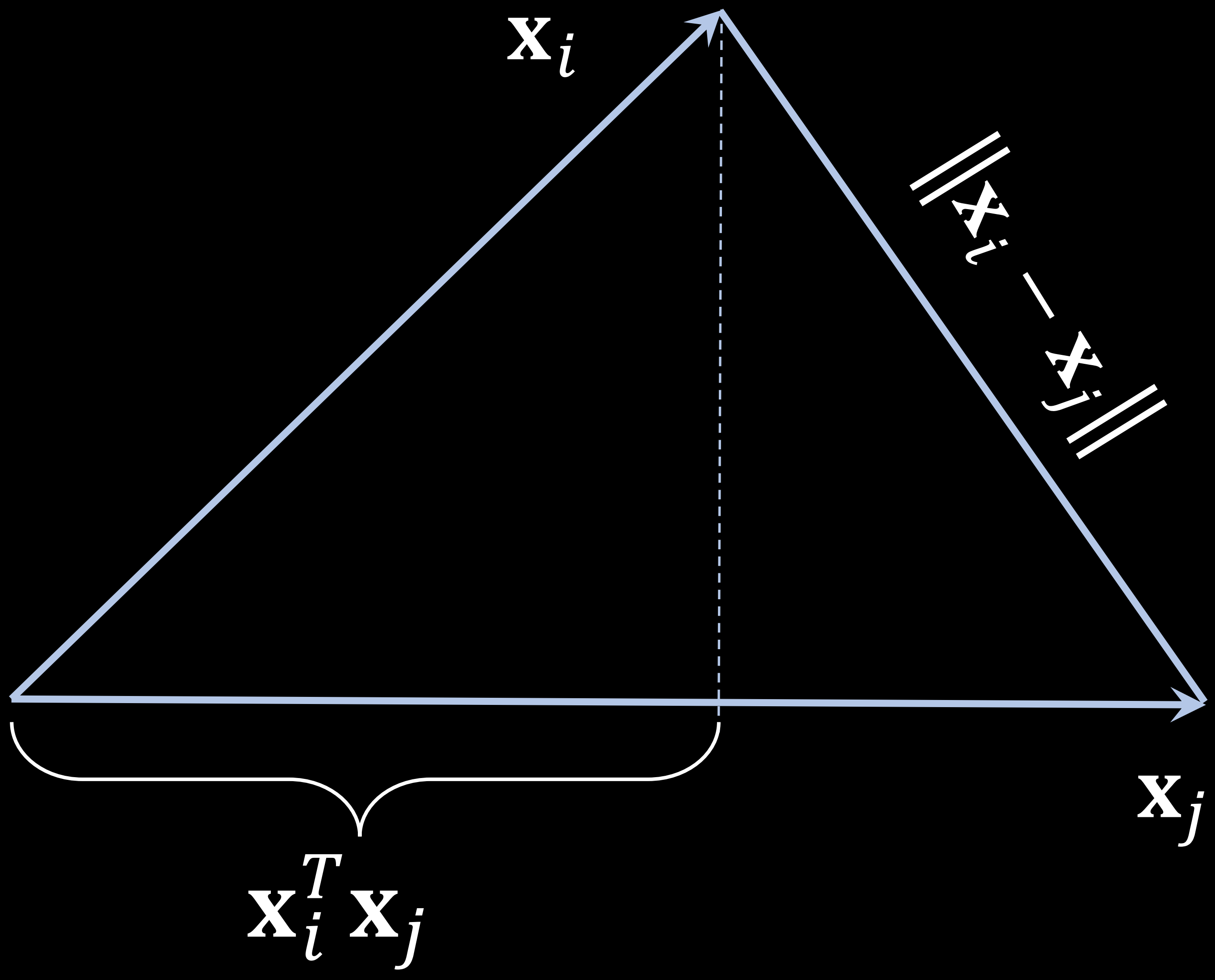
Fig. 1.19 illustrating the relationship between *inner product* and *Euclidean distance* as a measure of the **similarity between patterns**



the smaller the Euclidean distance $\|\mathbf{x}_i - \mathbf{x}_j\|$, the more similar the vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ and the larger the inner product $\mathbf{x}_i^T \mathbf{x}_j$ will be.

To put this relationship on a formal basis, we first normalize

Measure of Similarity Between Patterns

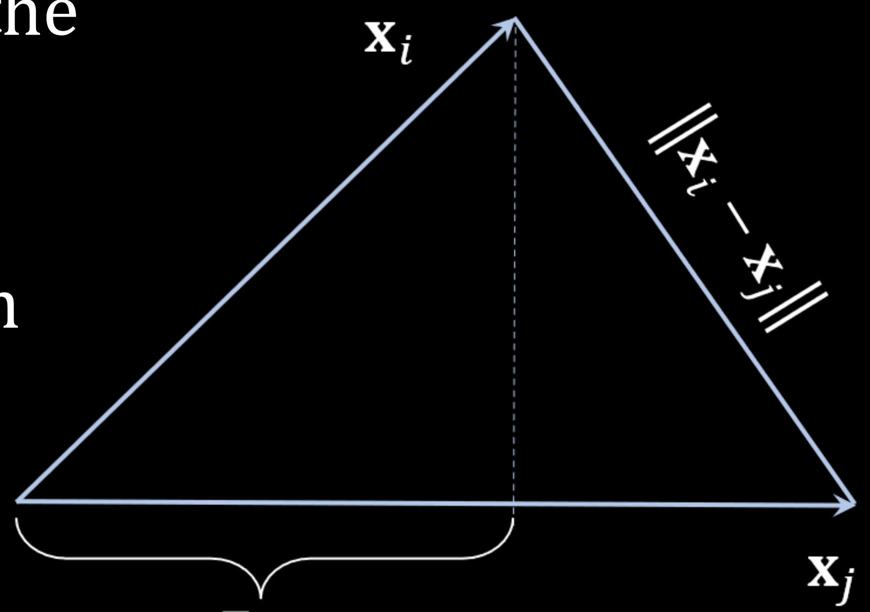


Measure of Similarity Between Patterns

- The smaller the Euclidean distance $\|\mathbf{x}_i - \mathbf{x}_j\|$, the more similar the vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ and the larger the inner product $\mathbf{x}_i^T\mathbf{x}_j$ will be.

- To put this relationship on a formal basis, we first normalize the vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ to have unit length

$$\|\mathbf{x}_i\| = \|\mathbf{x}_j\| = 1$$



And rewrite (1.23) as

$$d^2(\boldsymbol{x_i}, \boldsymbol{x_j}) = (\mathbf{x}_i - \mathbf{x}_j)^{\boldsymbol{T}}(\mathbf{x}_i - \mathbf{x}_j) = \mathbf{2} - \mathbf{2}\mathbf{x}_i^T\mathbf{x}_j \qquad (1.25)$$

(1.25) is the minimization of the Euclidean distance $d(\boldsymbol{x_i}, \boldsymbol{x_j})$ corresponds to maximization of the inner product $(\boldsymbol{x_i}, \boldsymbol{x_j})$ and, therefore, the similarity between the vectors $\mathbf{x}_i$ and $\mathbf{x}_j$

Measure of Similarity Between Patterns

- When the vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ are drawn from two different populations (pools) of data with means $\boldsymbol{\mu}_i$ and $\boldsymbol{\mu}_j$,

$$\boldsymbol{\mu}_i = E[\mathbf{x}_i] \quad \text{and } \boldsymbol{\mu}_j = E[\mathbf{x}_j] \tag{1.26}$$

- The square of the distance from $\mathbf{x}_i$ to $\mathbf{x}_j$ is defined by

$$d_{ij}^2 = (\mathbf{x}_i - \boldsymbol{\mu}_i)^T \Sigma^{-1} \left(\mathbf{x}_j - \boldsymbol{\mu}_j\right) \tag{1.27}$$

Where Σ is the covariance matrix given by

$$\boldsymbol{\Sigma} = \mathrm{E}[(\mathbf{x}_i - \boldsymbol{\mu}_i)(\mathbf{x}_i - \boldsymbol{\mu}_i)^T] = \mathrm{E}\left[\left(\mathbf{x}_j - \boldsymbol{\mu}_j\right)\left(\mathbf{x}_j - \boldsymbol{\mu}_j\right)^T\right] \tag{1.28}$$

Rules for Knowledge Representation – Rule 2

- **Rule 2.** Items to be categorized as separate classes should be given widely different representations in the network.

  The second rule is the exact opposite of Rule 1.

Rules for Knowledge Representation – Rule 3

- **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.
- Consider, for example, a radar application involving the detection of a target (e.g., aircraft) in the presence of clutter (i.e., radar reflections from undesirable targets such as buildings, trees, and weather formations).
- The detection performance of such a radar system is measured in terms of two probabilities:
  - ~ Probability of detection, defined as the probability that the system decides that a target is present when it is.
  - ~ Probability of false alarm, defined as the probability that the system decides that a target is present when it is not.

Rules for Knowledge Representation – Rule 4

Rule 4. Prior information and invariances should be built into the design of a neural network, to simplify the network design by not having to learn them. Rule 4 is particularly important because proper devotion to it results in a neural network with a specialized (restricted) structure. This is highly desirable for several reasons

1. Biological visual and auditory networks are known to be very specialized.

2. A neural network with specialized structure usually has a smaller number of free parameters available for adjustment than a fully connected network, requiring smaller data set for training, learns faster and generalizes better

3. The rate of information transmission through a specialized network (i.e., the network throughput) is accelerated.

4. The cost of building a specialized network is reduced

How to Build Prior Information into Neural Network Design

- To develop a specialized structure, a combination of two techniques can be employed:

1. **Restricting network architecture** through the use of local connections known as **receptive fields**

2. Constraining the choice of synaptic weights through weight-sharing

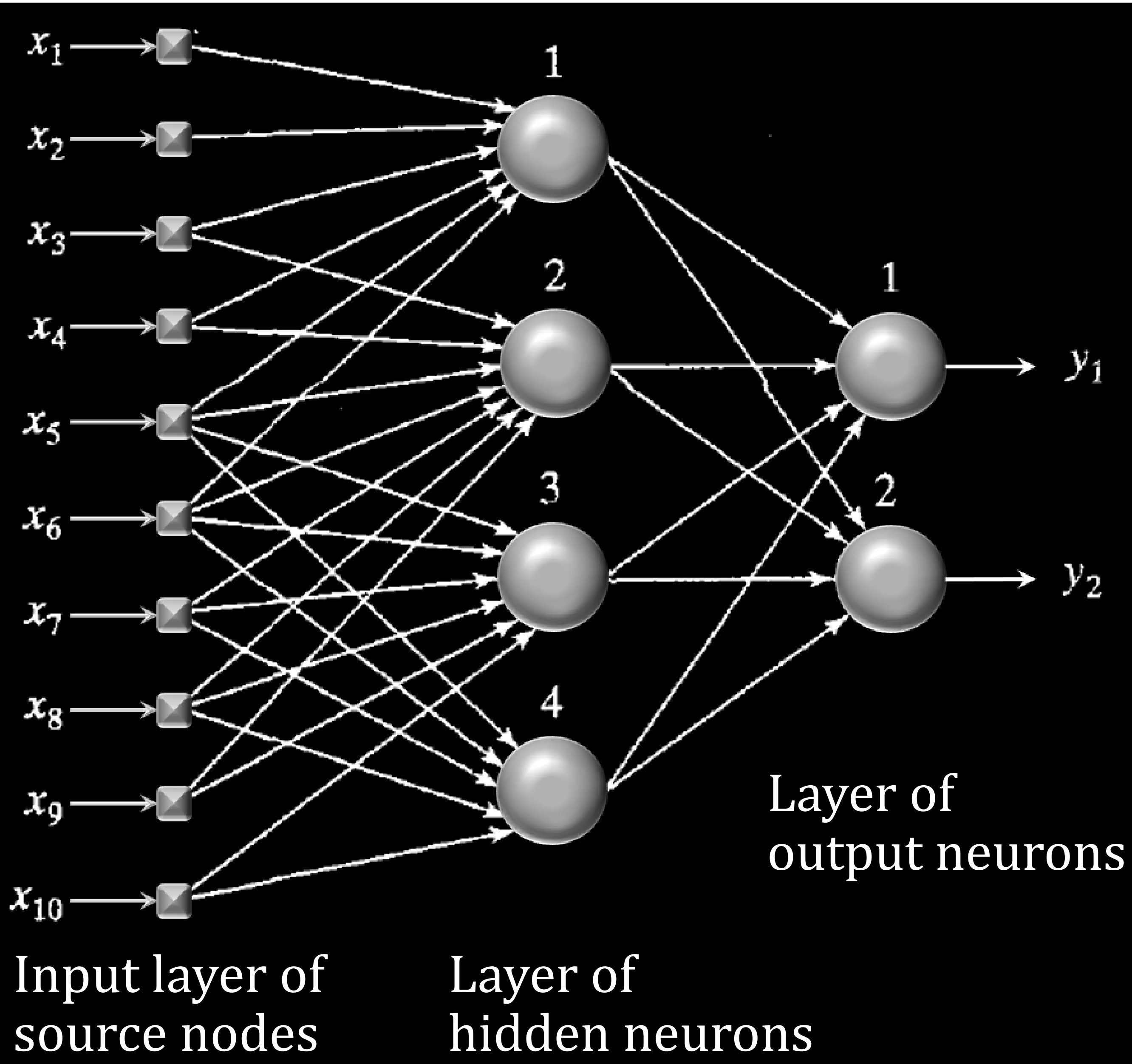
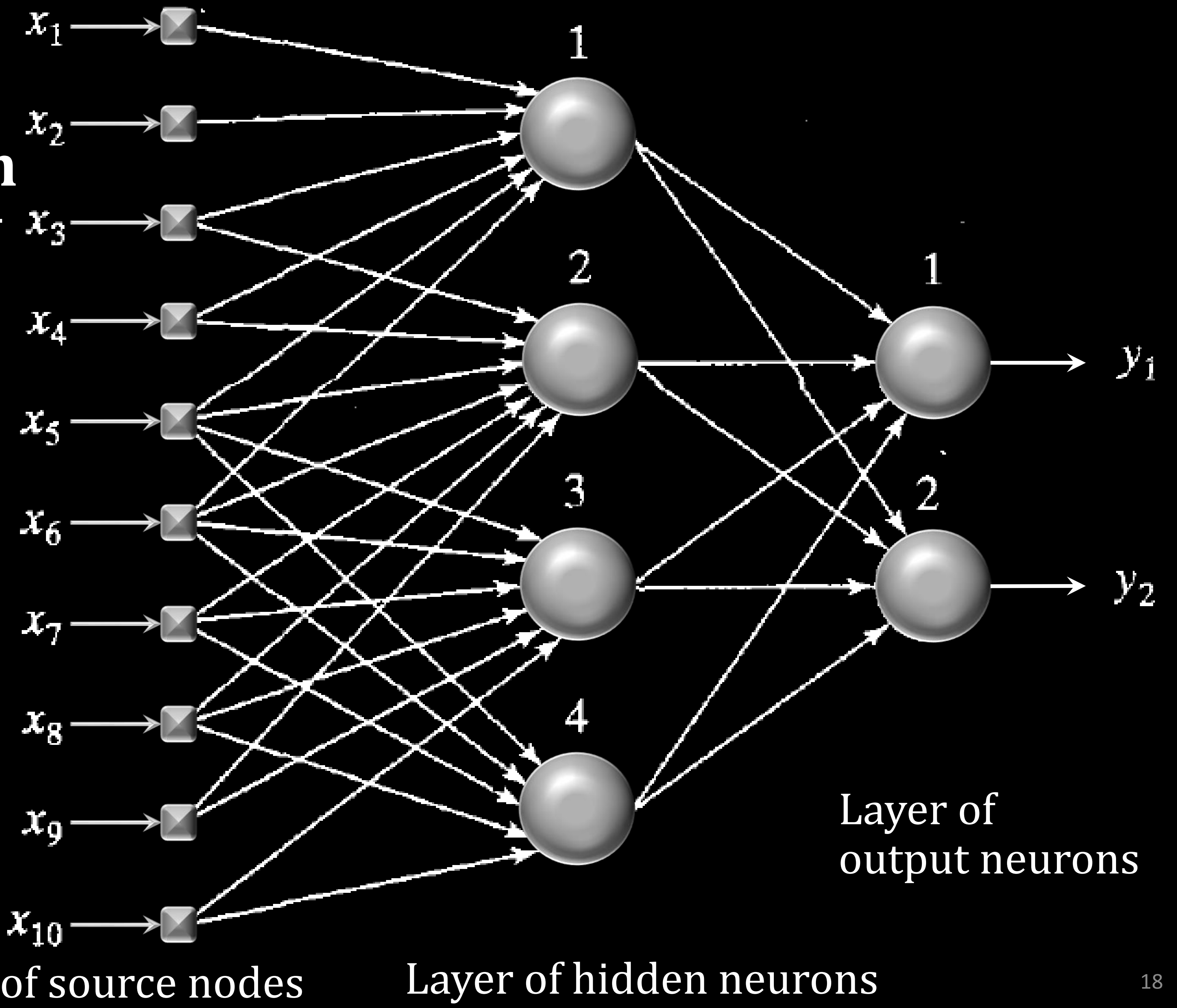


Fig. 1.20 Illustrating the combined use of a receptive field and weight-sharing. All four hidden neurons share the same set of weights for their synaptic connections

How to Build Prior Information into Neural Network Design

Fig. 1.20 Illustrating the combined use of a receptive field and weight-sharing. All four hidden neurons share the same set of weights for their synaptic connections



How to Build Prior Information into Neural Network Design

- we may express the induced local field of hidden neuron j in Fig. 1.20 as

$$v_j = \sum_{i=1}^{6} w_i \kappa_{i+j-1}, \qquad j = 1, 2, 3, 4 \qquad (1.29)$$

where $\{w_i\}_{i=1}^{6}$ are the weights of the 4 hidden neurons. Equation (1.29) is in the form of a convolution sum. Therefore, the feedforward network using local connections and weight-sharing in this manner is referred to as a convolutional network.

How to Build Invariances into Neural Network Design

- Consider the following physical phenomena:
  - When an object of interest rotates, the image of the object as perceived by an observer usually changes in a corresponding way.
  - In a coherent radar that provides amplitude as well as phase information about its surrounding environment, the echo from a moving target is shifted in frequency due to the Doppler effect that arises due to the radial motion of the target in relation to the radar.
  - The utterance from a person may be spoken in a soft or loud voice, and in a slow or quick manner.

How to Build Invariances into Neural Network Design

- A primary requirement of pattern recognition is to design a classifier that is invariant to (not affected by) such transformations applied to the classifier input.
- There are at least three techniques for rendering classifier-type neural networks invariant to transformations
  1. Invariance by Structure
  2. In variance by Training
  3. Invariant Feature Space

How to Build Invariances into Neural Network Design

- The third technique of creating an invariant classifier type neural network is illustrated in Fig. 1.21.

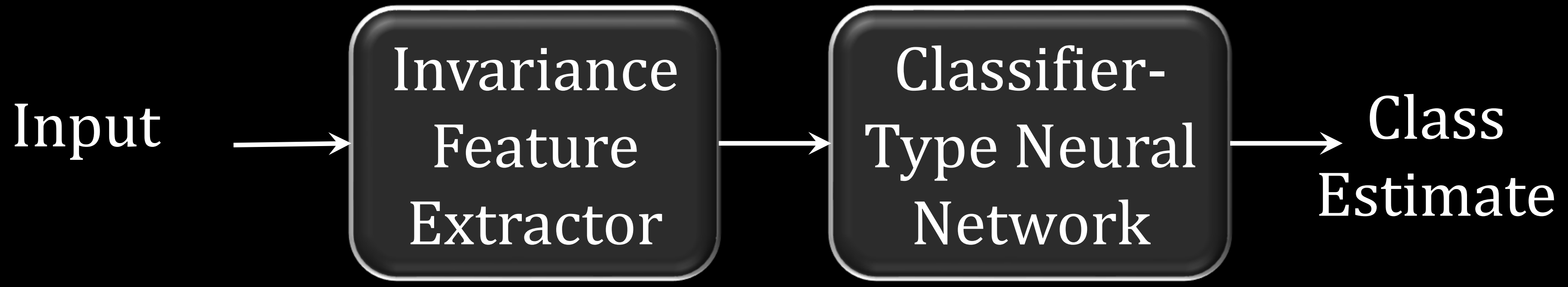


Fig. 1.21 Block diagram of an invariant feature-space type of system

- It is be possible to extract features that characterize the essential information content of an input data set, which is invariant to transformations of the input
- If such features are used, then the network as a classifier is relieved from the burden of having to delineate the range of transformations of an object with complicated decision boundaries

How to Build Invariances into Neural Network Design

- The use of an invariant feature space offers three distinct advantages.
  - First, the number of features applied to the network may be reduced to realistic levels.
  - Second, the requirements imposed on network design are relaxed.
  - Third, invariance for all objects with respect to known transformations is assured
- However, this approach requires prior knowledge of the problem for it to work

How to Build Invariances into Neural Network Design

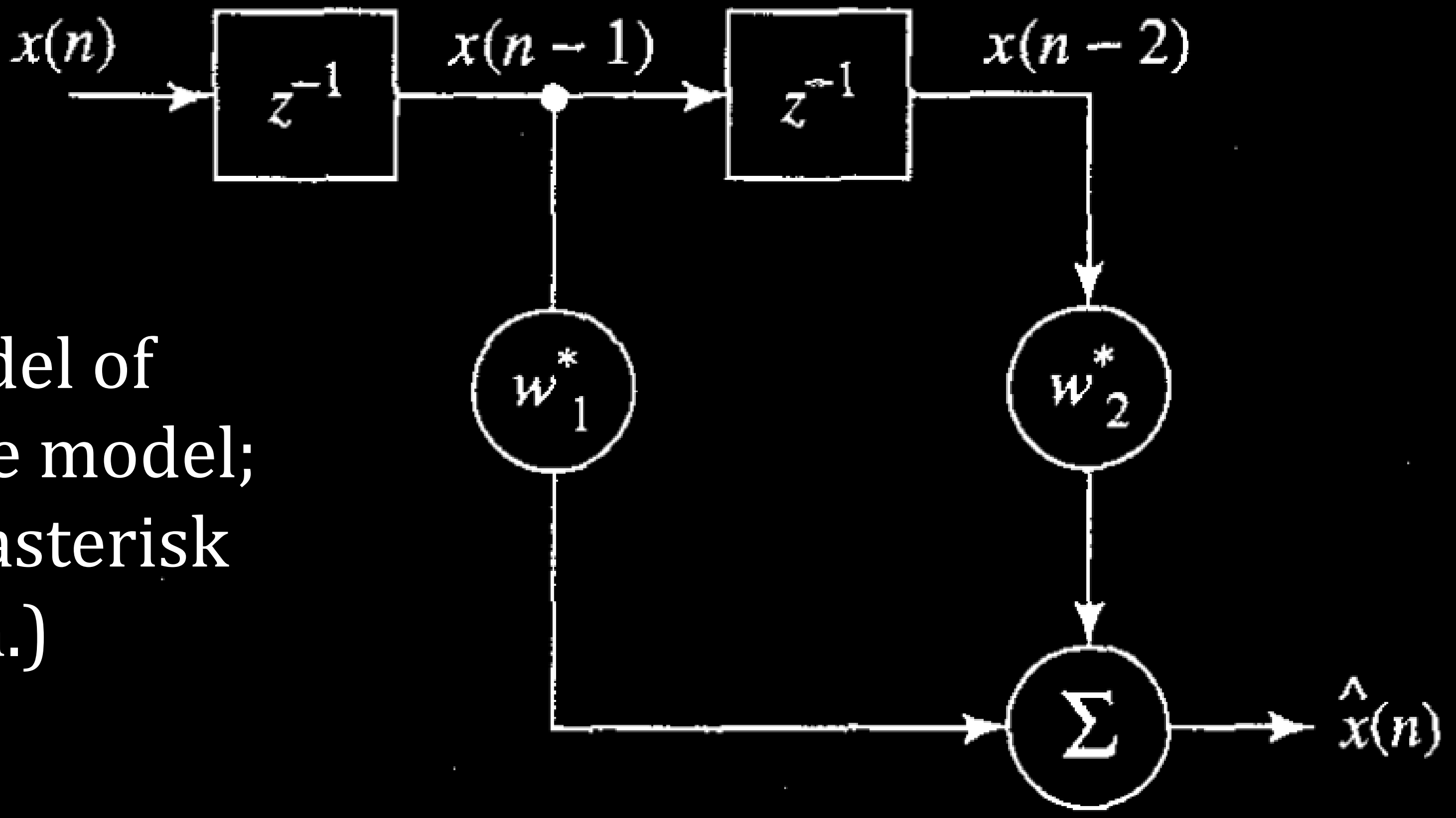
- To illustrate the idea of invariant-feature space, consider the example of a coherent radar system used for air surveillance
- Experimental studies have shown that such radar signals can be modeled as an autoregressive (AR) process
- An AR model is a special form of regressive model defined for complex-valued data by

$$x(n) = \sum_{i=1}^{M} a_i^* \, x(n-i) + e(n) \qquad (1.30)$$

- where $\{a_i\}_{i=1}^{M}$ are the AR coefficients, M is the model order, $x(n)$ is the input, and $e(n)$ is the error described as white noise.

How to Build Invariances into Neural Network Design

- The AR model of Eq. (1.30) is represented by a tapped-delay-line filter as illustrated in Fig. 1.22a for M = 2.

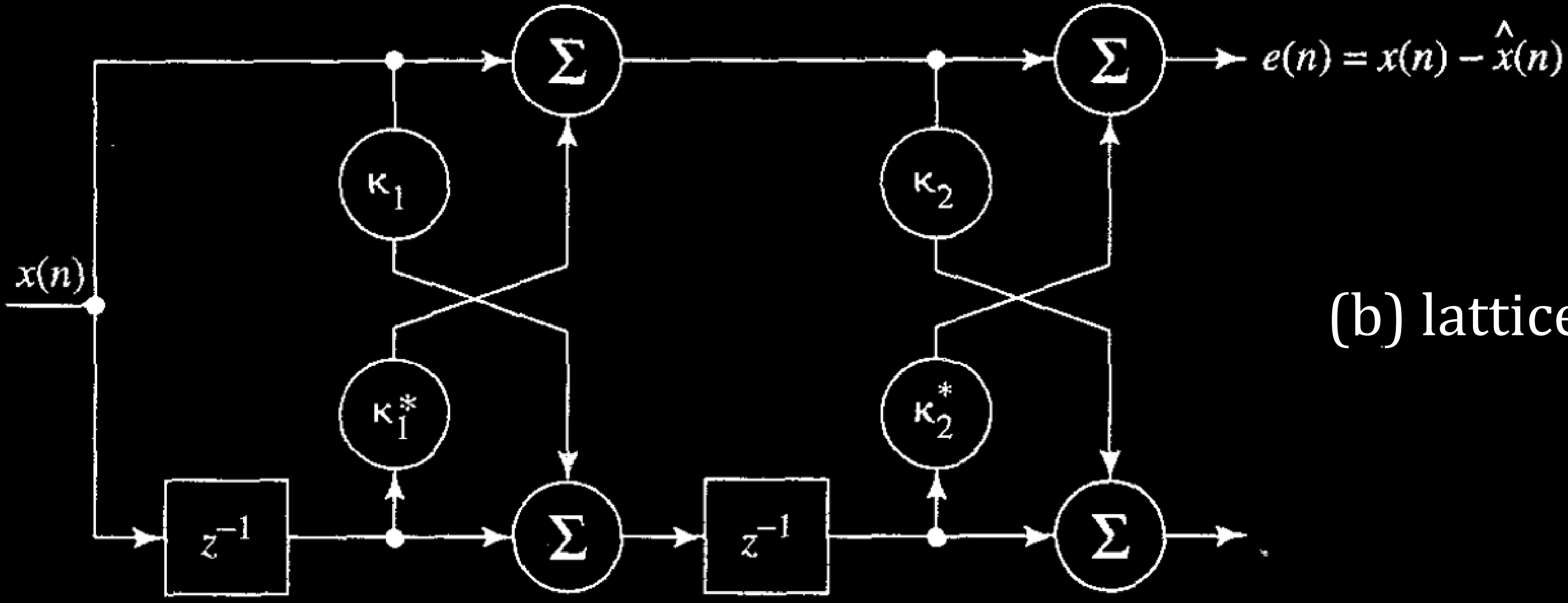


(a) tapped-delay-line model

Fig. 1.22 Autoregressive model of order 2: (a) tapped-delay-line model; (b) lattice filter model. (The asterisk denotes complex conjugation.)

How to Build Invariances into Neural Network Design

- The AR model of Eq. (1.30) is represented by a tapped-delay-line filter as illustrated in Fig. 1.22a for M = 2.



(b) lattice filter model

Fig. 1.22 Autoregressive model of order 2: (a) tapped-delay-line model; (b) lattice filter model. (The asterisk denotes complex conjugation.)

How to Build Invariances into Neural Network Design

- The two models depicted assume that the input $x(n)$ is complex valued, as in the case of a coherent radar, in which case the AR coefficients and the reflection coefficients are all complex valued.
- Hence, by defining a new set of reflection coefficients $\{\kappa_m'\}$ related to the set of ordinary reflection coefficients $\{\kappa_m\}$ computed from the input data as follows:

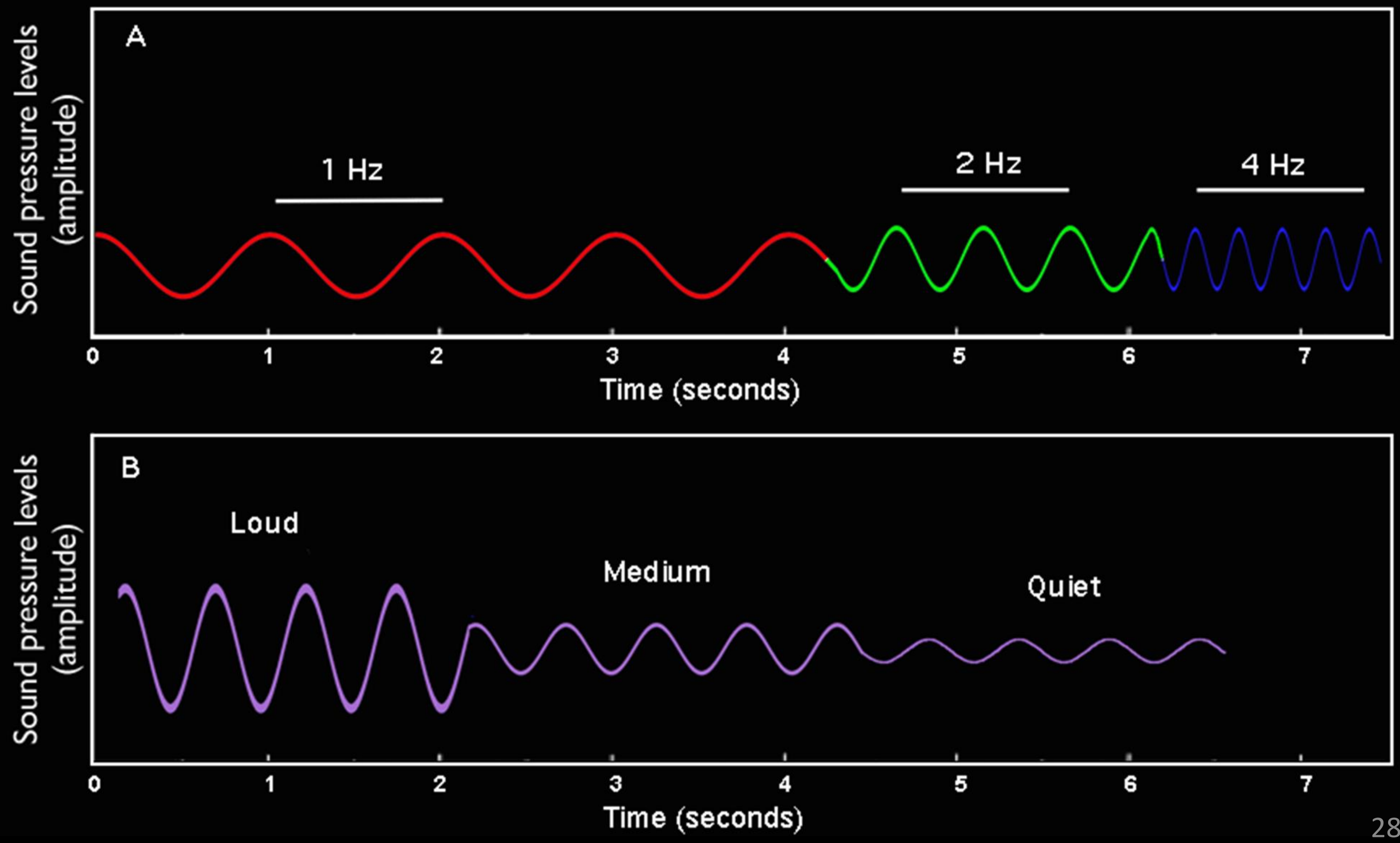
$$\kappa_m' = \kappa_m e^{-jm\theta} \quad \text{for } m = 1, 2, 3, \dots, M \tag{1.31}$$

where θ is the phase angle of the first reflection coefficient.

Knowledge Representation in Neural Networks

- An example of knowledge representation in a neural network is found in the biological sonar system of echo-locating bats.
- Bats use their mouth to broadcast short-duration FM sonar signals and uses its auditory system as the sonar receiver

- Echoes from targets of interest are represented in auditory system by activity of neurons that are selective to different combinations of acoustic parameters



Neural Dimensions of the Bat's Auditory Representation

- There are three principal neural dimensions of the bat's auditory representation

1. Echo frequency is encoded by the "place" originating in frequency map of the cochlea;

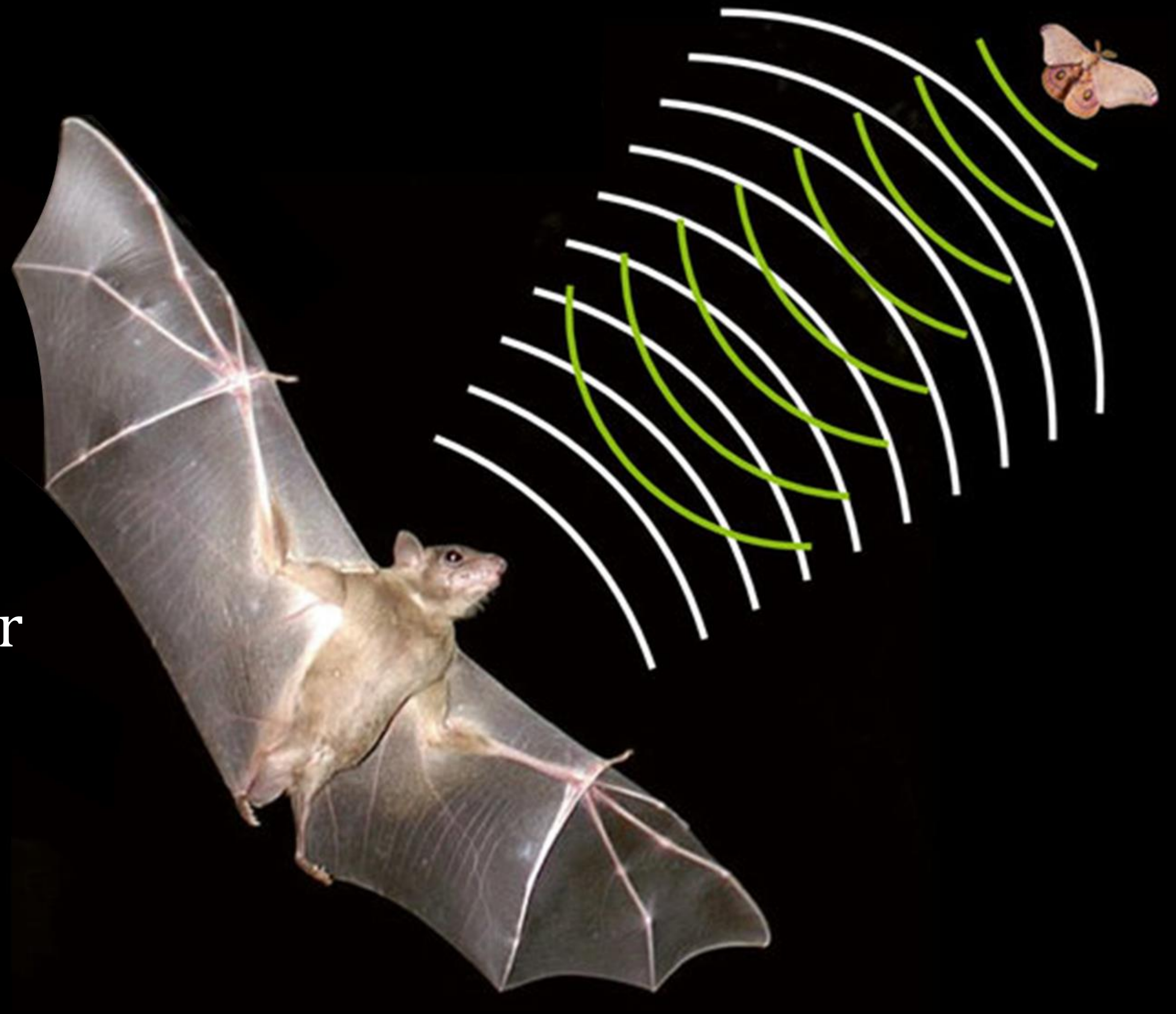
   It is preserved in auditory pathway as an arrangement of neurons tuned to different frequencies,

2. Echo amplitude, which is encoded by other neurons with different dynamic ranges;

   Both amplitude and number of discharges per stimulus are preserved.

3. Echo delay, which is encoded through neural computations that produce delay-selective responses;

   It is manifested as target range tuning.



(17) (PDF) Q&A: Cognitive ethology - inside the minds of other species (researchgate.net)

1.8 Artificial Intelligence and Neural Networks

- The goal of artificial intelligence (AI) is the development of paradigms or algorithms that require machines to perform cognitive tasks, at which humans are currently better.
- An AI system must be capable of doing three things:

  1) store knowledge,

  2) Apply the knowledge stored to solve problems, and

  3) acquire new knowledge through experience.
- An AI system has three key components:
  - representation,
  - reasoning, and
  - learning
- See Fig. 1.24.

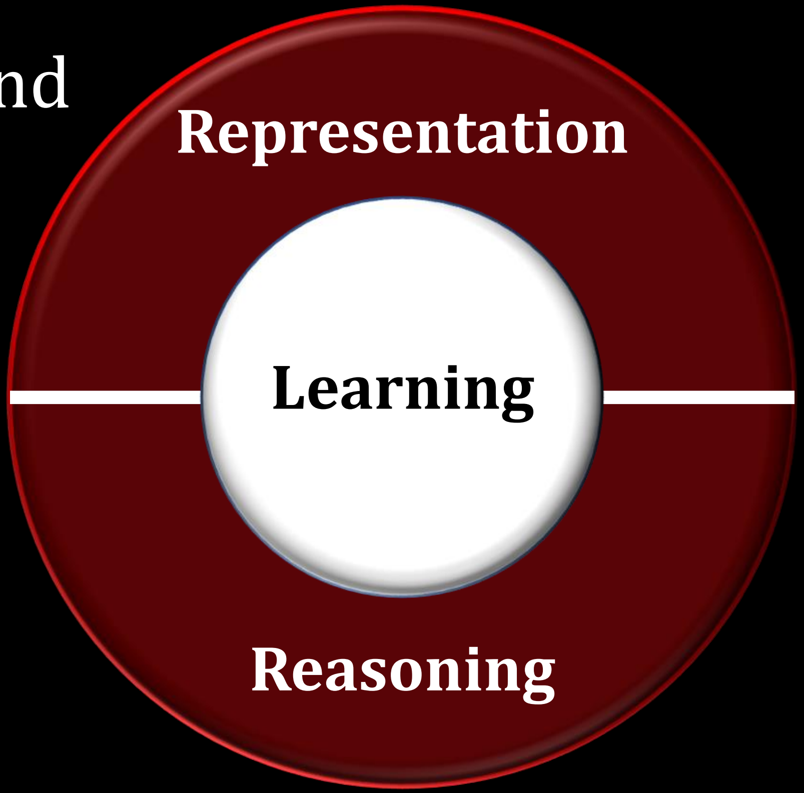


Fig. 1.24 Illustrating the three key components of an AI system.

Key Components of AI Systems

1. Representation

- The most distinctive feature of AI is the pervasive use of a language of symbol structures to represent both general knowledge about a **problem domain** of interest and specific knowledge about the **solution** to the problem.
- The clarity and familiarity of the symbolic representation makes AI easy to understand and well suited for human-machine communication.
- **Knowledge** is represented as a static collection of facts, with a small set of general procedures used to manipulate the facts.

Key Components of AI Systems

2. Reasoning

- Reasoning is the ability to solve problems.
- A system qualifies as a reasoning system if it satisfies certain conditions
  - ~ The system must be able to express and solve a broad range of problems and problem types
  - ~ The system must be able to make explicit and implicit information known to it.
  - ~ The system must have a control mechanism that determines which operations to apply to a particular problem, when a solution to the problem has been obtained or when further work on the problem should be terminated.

Key Components of AI Systems

3. Learning

- In the simple model of machine learning depicted in Fig. 1.25,
  - ~ The environment supplies some information to a learning element.
  - ~ The learning element then uses this information to make improvements in a knowledge base, and finally
  - ~ The performance element uses the knowledge base to perform its task.
  - ~ The machine therefore operates by guessing, and then receiving feedback from the performance element.
  - ~ The feedback mechanism enables the machine to evaluate its hypotheses and revise them if necessary.

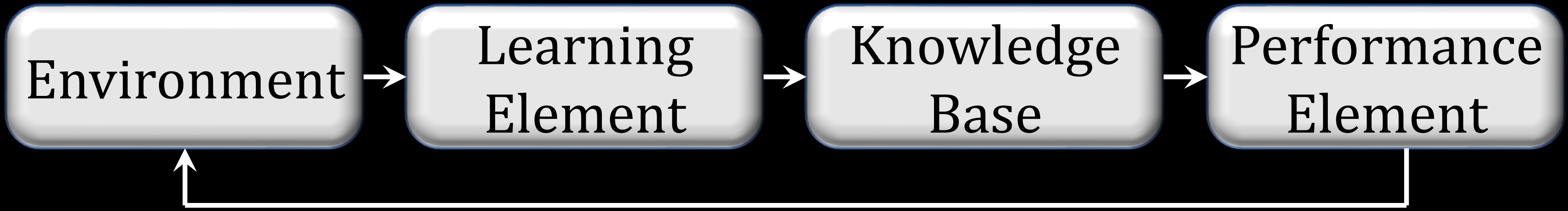


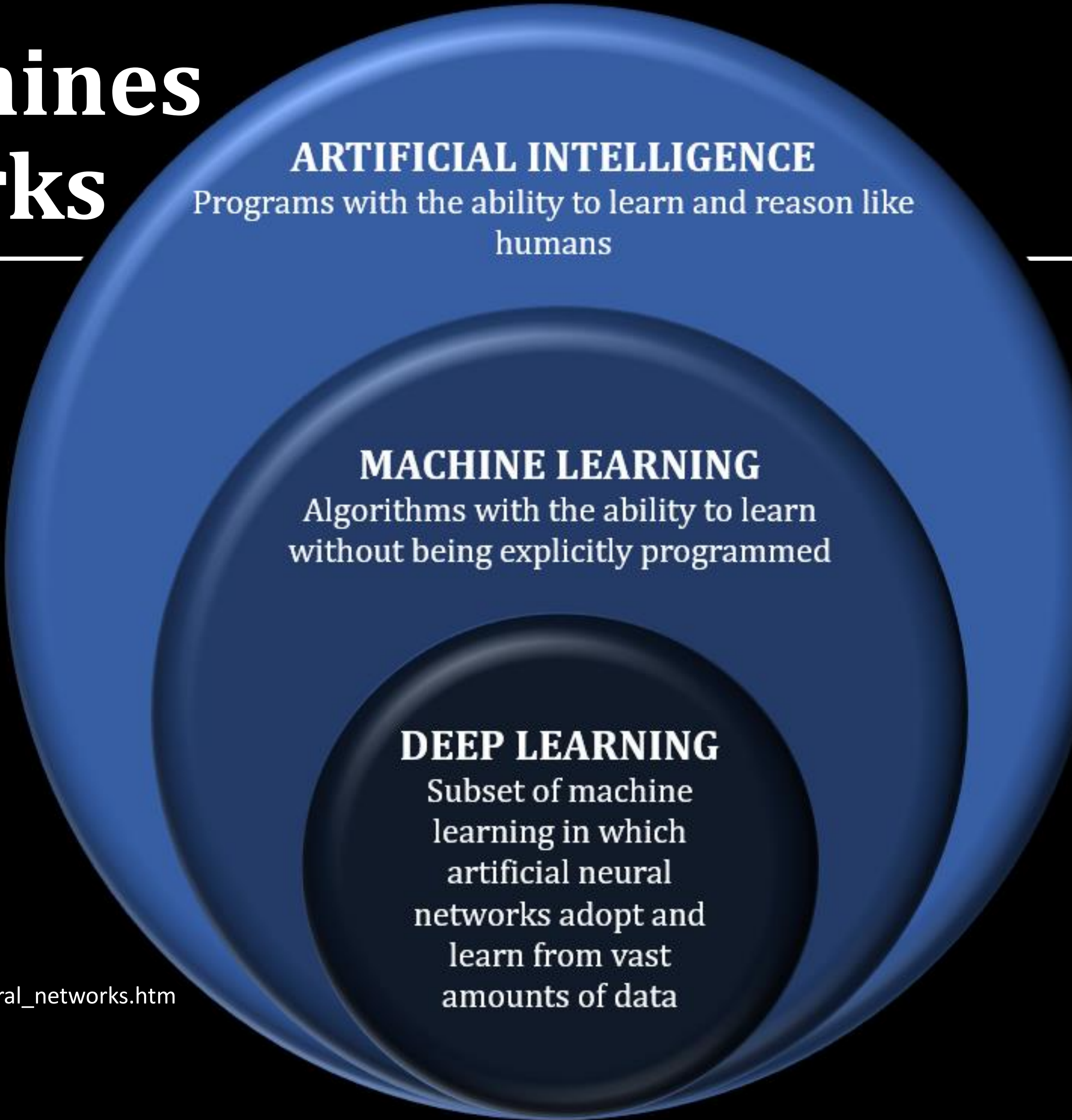
Fig. 1.25 Simple model of machine learning.

Key Components of AI Systems

- Machine learning involves two different kinds of information processing:
  - In **inductive** information processing, general patterns and rules are determined from raw data and experience. Similarity-based learning uses induction.
  - In **deductive** information processing, general rules are used to determine specific facts. Proof of a theorem uses deduction from known axioms and other existing theorems.
  - Explanation-based learning uses both induction and deduction.
- In an **expert system**, the compiled experience of a field expert is duplicated, and the experience is directed towards solving a problem.

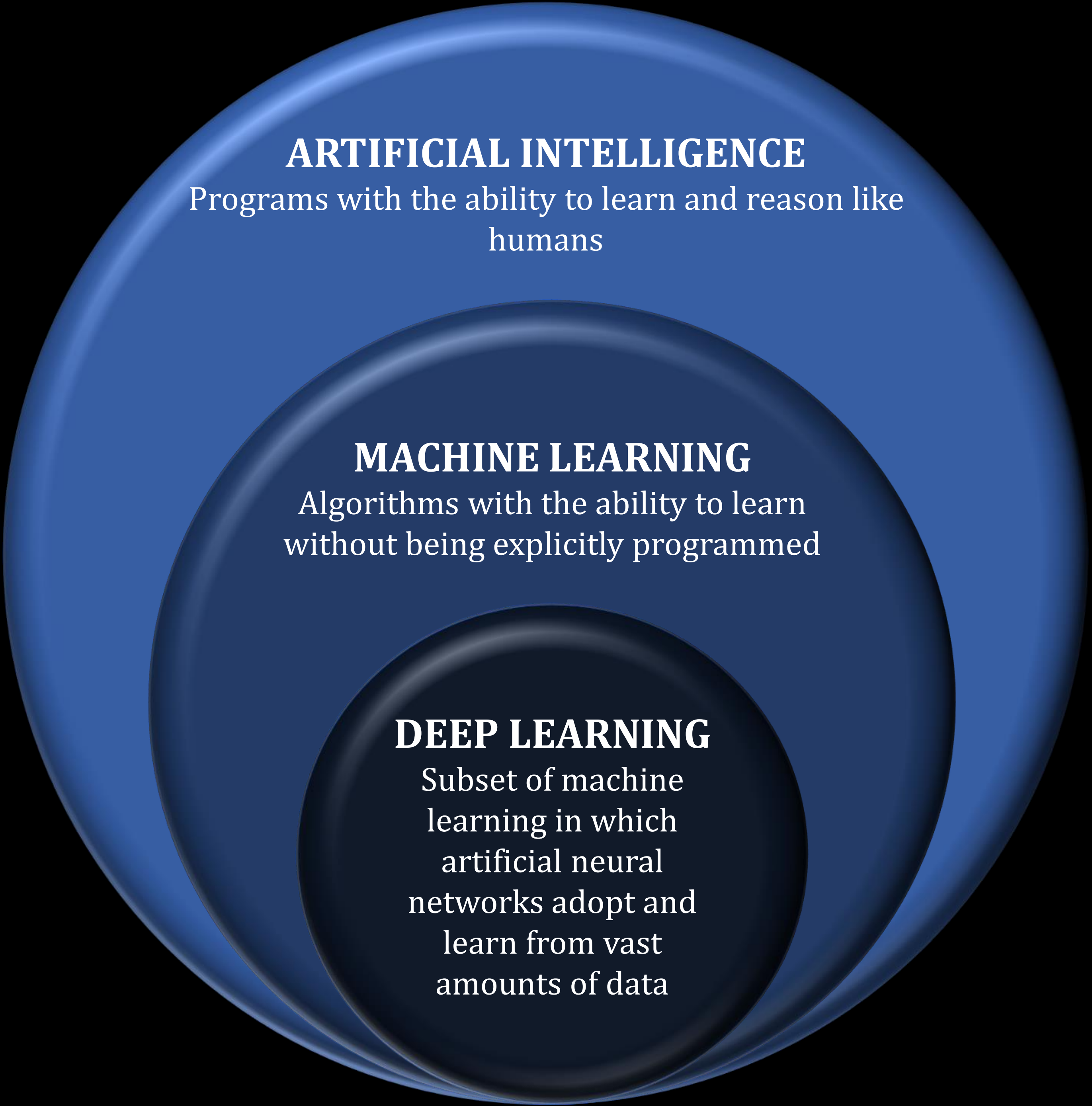
Comparing AI Machines to Neural Networks

- To compare AI machines and Neural Networks, three subdivisions are considered
  1. level of explanation,
  2. style of processing, and
  3. representational structure



https://www.tutorialspoint.com/artificial_intelligence/artificial_intelligence_neural_networks.htm

Comparing AI Machines to Neural Networks



Comparing AI Machines to Neural Networks

1. Level of Explanation.

AI	NN
In classical AI, the emphasis is on building symbolic representations. Modern AI assumes the existence of mental representations and it models cognition as the sequential processing of symbolic representations	The emphasis in neural networks is on the development of parallel distributed processing (PDP) models with large number of neurons, neurons interacts with each other. Moreover, neural networks place great emphasis on neurobiological explanation of cognitive phenomena.

Comparing AI Machines to Neural Networks

2. Processing Style

AI	NN
In classical AI, the processing is sequential, as in typical computer programming. The operations are performed in a step-by-step manner inspired from the von Neumann machine	Parallelism is conceptually essential to the processing of information Massive parallelism gives neural networks a remarkable robustness with the computation spread over many neurons Noisy or incomplete inputs may still be recognized, a damaged network may function satisfactorily, and learning does not have to be perfect

Comparing AI Machines to Neural Networks

2. Representational Structure

AI

The expressions of classical AI are generally complex, built in a systematic fashion from simple symbols

AI is the formal manipulation of a language of algorithms and data representations in a top-down fashion

NN

Neural networks are more adequate in dealing with cognition and linguistics

Neural networks, are parallel distributed processors with a natural ability to learn and which usually operate in a bottom-up fashion.

Comparing AI Machines to Neural Networks

- For the implementation of **cognitive** tasks, a more potentially useful approach is to build *structured connectionist* models or *hybrid systems* that integrate AI and NN together.
- Such combined AI and NN systems would combine the desirable features of adaptivity, robustness, and uniformity offered by neural networks with the representation, inference, and universality that are inherent features of symbolic AI

Cognition: The mental action or process of acquiring knowledge and understanding through thought, experience, and the senses.

Comparing AI Machines to Neural Networks

- The reasons for the extraction of rules from neural networks
  - ~ To validate neural network components in software systems by making the internal states of the neural network accessible and understandable to users.
  - ~ To improve the generalization performance of neural networks by (1) identifying regions of the input space where the training data are not adequately represented, or (2) indicating the circumstances where the neural network may fail to generalize.
  - ~ To discover salient features of the input data for data exploration (mining).
  - ~ To provide a means for traversing the boundary between the connectionist and symbolic approaches to the development of intelligent machines.
  - ~ To satisfy the critical need for safety in a special class of systems where safety is a mandatory condition.

1.9 Historical Notes

- The modern era of neural networks began with the pioneering work of McCulloch and Pitts (1943). He worked on representation of an event in the nervous system.
- In 1943, in their classic paper, McCulloch and Pitts describe a logical calculus of neural networks.
- Their formal model of a neuron was assumed to follow an "all-or-none" law.
- McCulloch and Pitts showed that such a network would compute any computable function.
- This led to the formal birth of neural networks and of artificial intelligence disciplines.

1.9 Historical Notes

- The 1943 paper by McCulloch and Pitts influenced von Neumann to the construction of the EDVAC (Electronic Discrete Variable Automatic Computer) that developed out of the ENIAC (Electronic Numerical Integrator and Computer)
- The ENIAC was the first general purpose electronic computer, which was built at the Moore School of Electrical Engineering of the University of Pennsylvania from 1943 to 1946.
- In 1961, second edition of Wiener's famous book Cybernetics was published, adding new material on learning and self-organization in addition to control, communications, and statistical signal processing.
- 30 yesrs later, Hopfield brought the linkage between statistical mechanics and learning systems to full fruition.

1.9 Historical Notes

- The next major development in neural networks came in 1949 with the publication of Hebb's book *The Organization of Behavior*, in which an explicit statement of a physiological learning rule for synaptic modification was presented for the first time.
- Hebb’s famous postulate of learning, which states that the effectiveness of a variable synapse between two neurons is increased by the repeated activation of one neuron by the other across that synapse.
- Hebb's book inspired the development of computational models of learning and adaptive systems
- Rochester, Holland, Haibt, and Duda (1956) uses computer simulation to test a well-formulated neural theory based on Hebb's postulate of learning;

1.9 Historical Notes

- Uttley (1956) demonstrated that a neural network with modifiable synapses may learn to classify simple sets of binary patterns into corresponding classes
- Ashby's book (1952), Design for a Brain: The Origin of Adaptive Behavior, was published where adaptive behavior was said to be learned but not inborn and that through learning, behavior of animals (systems) changes for the better.
- In 1954, Minsky wrote a '"neural network" doctorate thesis at Princeton University, which was entitled "Theory of Neural-Analog Reinforcement Systems and Its Application to the Brain-Model Problem"
- In 1961, an excellent early paper by Minsky on AI entitled "Steps Toward Artificial Intelligence," was published; this latter paper contains a large section on what is now termed neural networks.
- In 1967 Minsky's book, Computation: Finite and Infinite Machines, was published where the concept of automata theory was introduced.

1.9 Historical Notes

- Also in 1954, the idea of a nonlinear adaptive filter was proposed by Gabor, one of the early pioneers of communication theory, and the inventor of holography.
- In the 1950s work on associative memory was initiated by Taylor (1956). This was followed by the introduction of the learning matrix by Steinbuch (1961);
- In 1969, an elegant paper on nonholographic associative memory by Willshaw, Buneman, and Longuet-Higgins was published.
- The von Neumann architecture, basic to the design of a digital computer, is named in his honor.

1.9 Historical Notes

- In 1958, the pattern recognition problem was introduced by Rosenblatt on his work on the perceptron, a novel method of supervised learning.
- In 1960, Widrow and Hoff introduced the least mean-square (LMS) algorithm and used it to formulate the Adaline (adaptive linear element).
- The difference between the perceptron and Adaline lies in the training procedure.
- In 1967, Amari used the stochastic gradient method for adaptive pattern classification.
- In 1965, Nilsson's book, Learning Machines, was published covering linearly separable patterns in hypersurfaces.
- The book by Minsky and Papert (1969) demonstrated that there are fundamental limits on what single-layer perceptrons can compute.
- In a brief section on multilayer perceptrons, they stated that any of the limitations of single-layer perceptrons could not be overcome in the multilayer version.

1.9 Historical Notes

- An important problem encountered in the design of a multilayer perceptron is the problem of assigning credit to hidden neurons in the network.
- There were three reasons for the 10 years delay about the solution of perceptron credit assignment problem:
    1. There were no personal computers or workstation for experimentation.
    2. Psychological and financial reasons. The 1969 monograph by Minsky and Papert discouraged work on perceptrons or agencies to support the work on them.
    3. The analogy between neural networks and lattice spins was premature.

1.9 Historical Notes

- In the 1970s, self-organizing maps using competitive learning was introducrd
- In 1976 Willshaw and von der Malsburg published the first paper on the formation of self-organizing maps in the brain.
- Grossberg (1980), built his earlier work on competitive learning (Grossberg, 1972, 1976a, b), adaptive resonance theory (ART).
- In 1982, Hopfield formulated recurrent networks with symmetric synaptic connections with isomorphism between such a recurrent network and an *Ising model*.

1.9 Historical Notes

- In 1983, Cohen and Grossberg established a general principle for assessing the stability of a content-addressable-memory that includes the continuous-time version of the Hopfield network.
- Kohonen's paper on self-organizing maps (Kohonen, 1982) using a one- or two-dimensional lattice structure, received far more attention in an analytic context.

1.9 Historical Notes

- In 1983, Kirkpatrick, Gelatt, and Vecchi described a new procedure called simulated annealing, for solving combinatorial optimization problems.
- The idea of simulated annealing was later used by Ackley, Hinton, and Sejnowski (1985) Boltzmann machine, which was the first successful realization of a multilayer neural network.
- The Boltzmann machine also laid the basis for the subsequent development of sigmoid belief networks by Neal (1992), which accomplished two things:

  1) significant improvement in learning, and
  2) Linking neural networks to belief networks

- A paper by Barto, Sutton, and Anderson on reinforcement learning was published in 1983.
- In 1996, the book Neurodynamic Programming by Bertsekas and Tsitsiklis was published.

1.9 Historical Notes

- In 1984 Braitenberg's book, Vehicles: Experiments in Synthetic Psychology, was published, focusing on goal-directed, self-organized performance.
- In 1986 the development of the back-propagation algorithm was reported by Rumelhart, Hinton, and Williams (1986).
- In 1986, the celebrated two-volüme book, Parallel Distributed Processing: Explorations in the Microstructures of Cognition, edited by Rumelhart and McClelland, was published.
- In 1988 Linsker described a new principle for self-organization in a perceptual network (Linsker, 1988a).
- The application of information theory to the blind source separation problem by Bell and Sejnowski (1995) has prompted many researchers to explore other information-theoretic models for solving a broad class of problems known collectively as **blind deconvolution**.

1.9 Historical Notes

- Also, in 1988, Broomhead and Lowe described a procedure for the design of layered feedforward networks using radial basis functions (RBF), which provide an alternative to multilayer perceptrons.
- In 1989, Mead's book, Analog VLSI and Neural Systems, was published, which provided an unusual mix of concepts drawn from neurobiology and VLSI technology.
- In the early 1990s, Vapnik and coworkers invented a computationally powerful class of supervised learning networks, called **Support Vector Machines**, for solving pattern recognition, regression, and density estimation problems.

1.9 Historical Notes

- According to Freeman, patterns of neural activity are not imposed from outside the brain; rather, they are constructed from within.
- Perhaps more than any other publication, the 1982 paper by Hopfield and the 1986 two-volume book by Rumelhart and McLelland were the most influential publications responsible for the resurgence of interest in neural networks in the 1980s.
- Neural networks have certainly come a long way from the early days of McCulloch and Pitts.
- Indeed, they have established themselves as an interdisciplinary subject with deep roots in the neurosciences, psychology, mathematics, the physical sciences, and engineering.