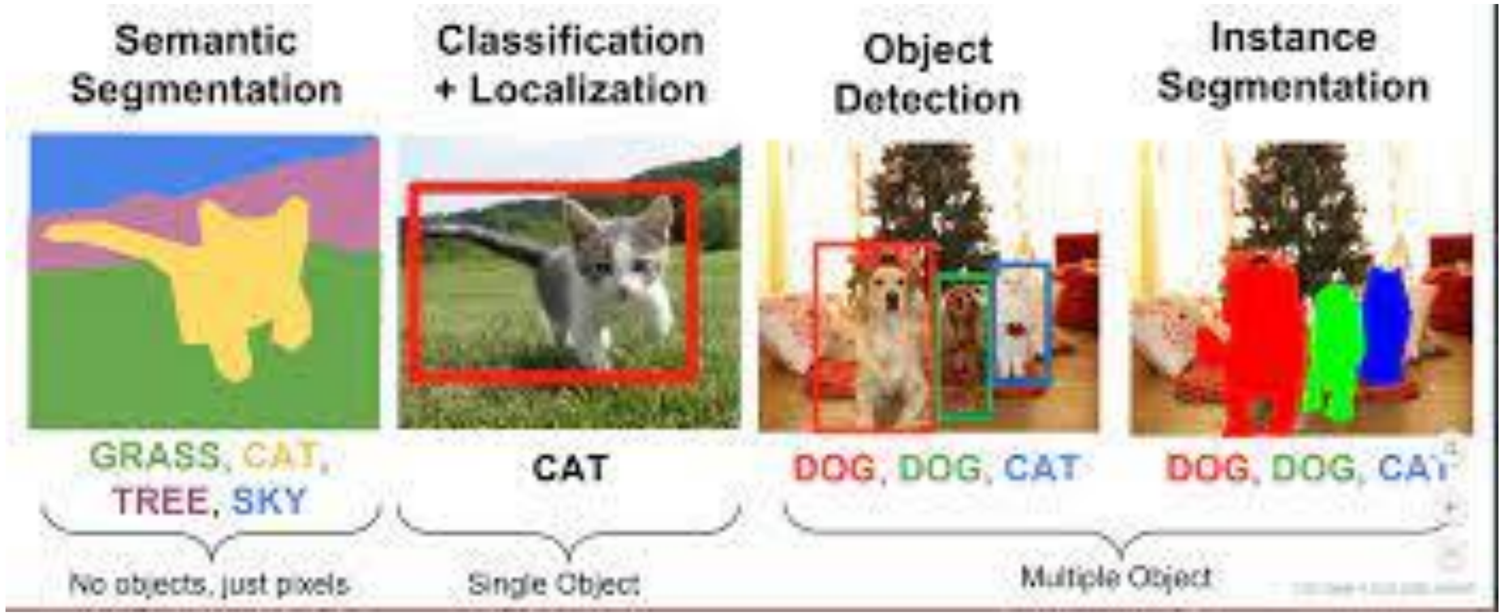
Image Processing and Pattern Recognition (IPPR)

Chapter 9:Object Recognition



Basanta Joshi, PhD

Asst. Prof., Depart of Electronics and Computer Engineering

Member, Laboratory for ICT Research and Development (LICT)

Institute of Engineering

basanta@ioe.edu.np

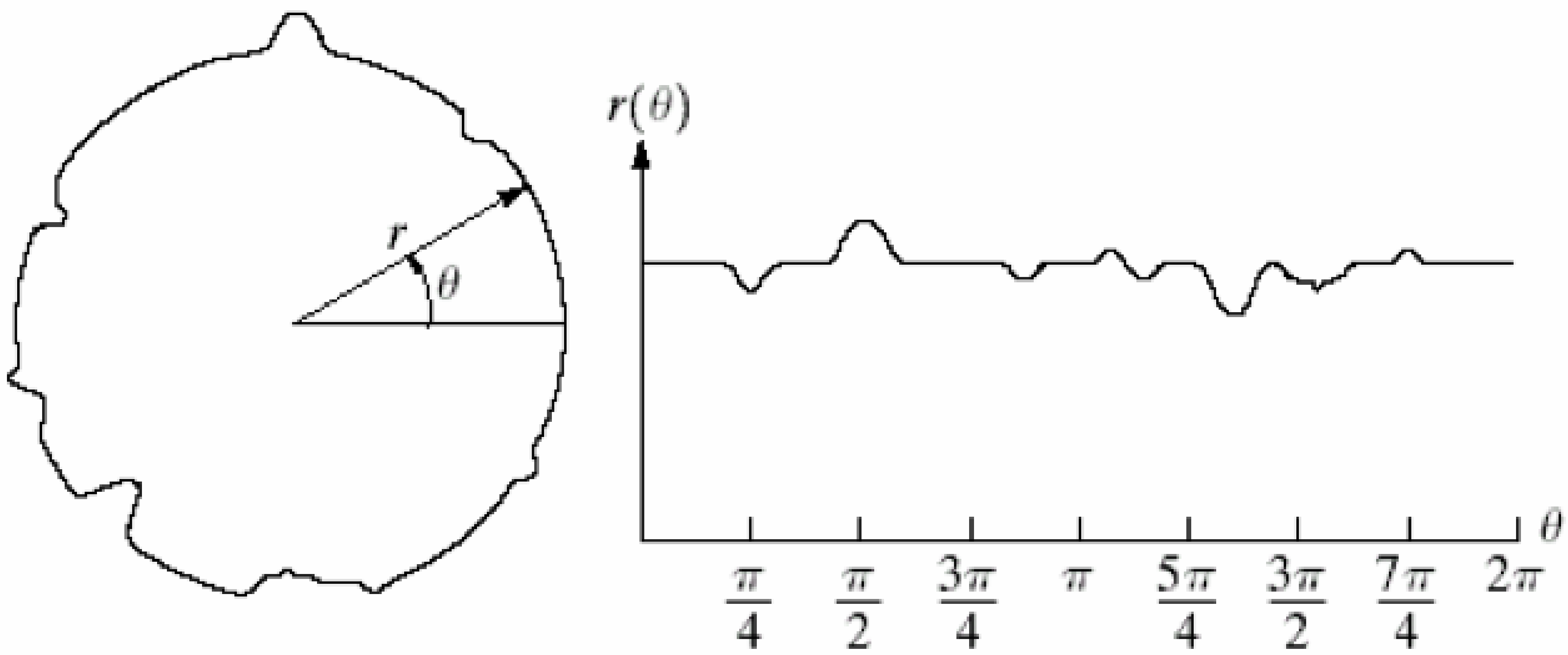
http://www.basantajoshi.com.np

https://scholar.google.com/citations?user=iocLiGcAAAAJ
https://www.researchgate.net/profile/Basanta_Joshi2

Pattern and Pattern Classes

- A *pattern* is an *arrangement of descriptors*, such as those discussed in Chapter 11.
- The name feature is used often in the pattern recognition literature to denote a descriptor.
- A pattern class is a family of patterns that share some common properties.

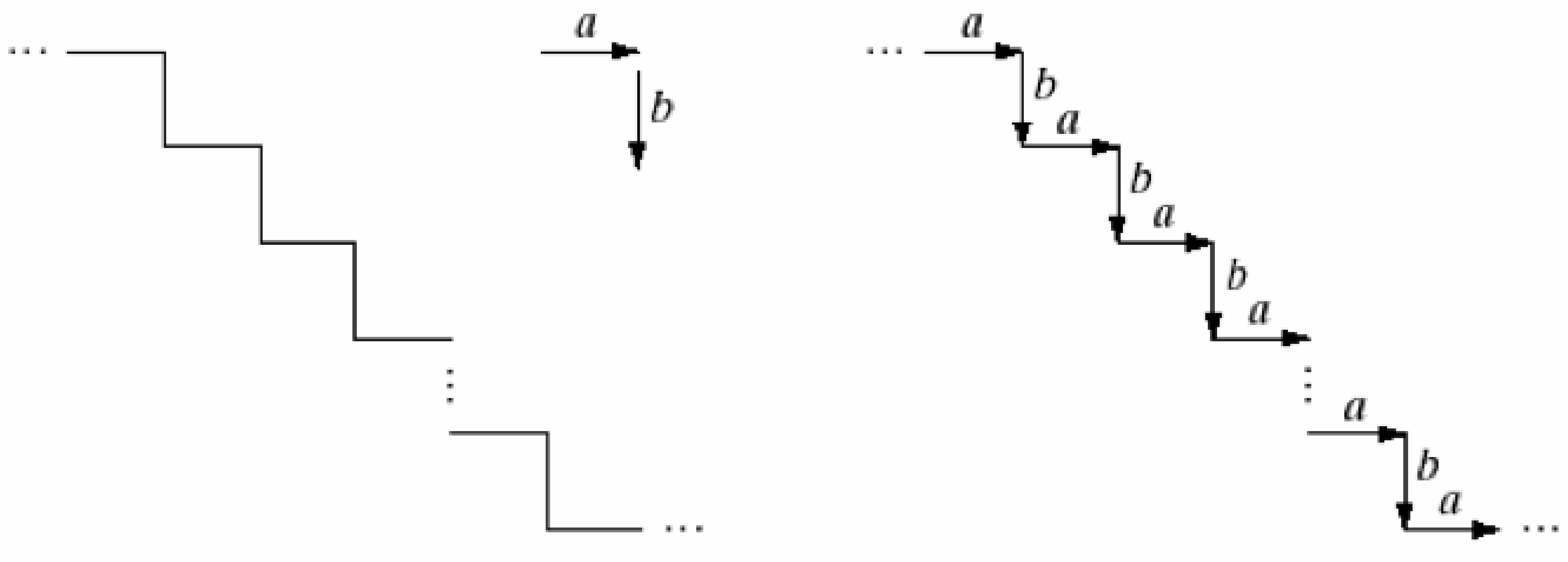
Signature



a b

FIGURE 12.2 A noisy object and its corresponding signature.

String Descriptors



a b

FIGURE 12.3 (a) Staircase structure. (b) Structure coded in terms of the primitives *a* and *b* to yield the string description ... *ababab* ... .

Satellite Image

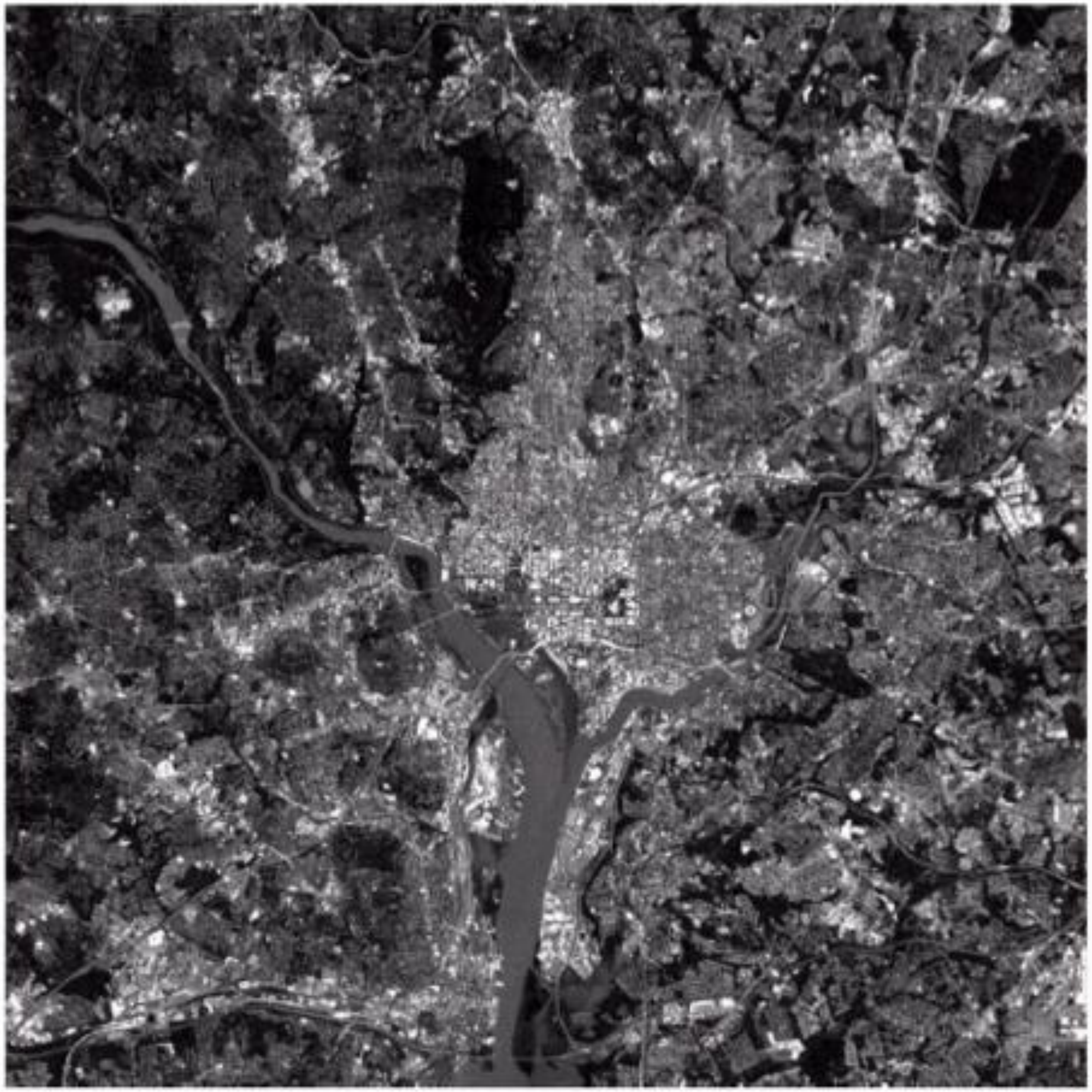


FIGURE 12.4 Satellite image of a heavily built downtown area (Washington, D.C.) and surrounding residential areas. (Courtesy of NASA.)

Image tree description

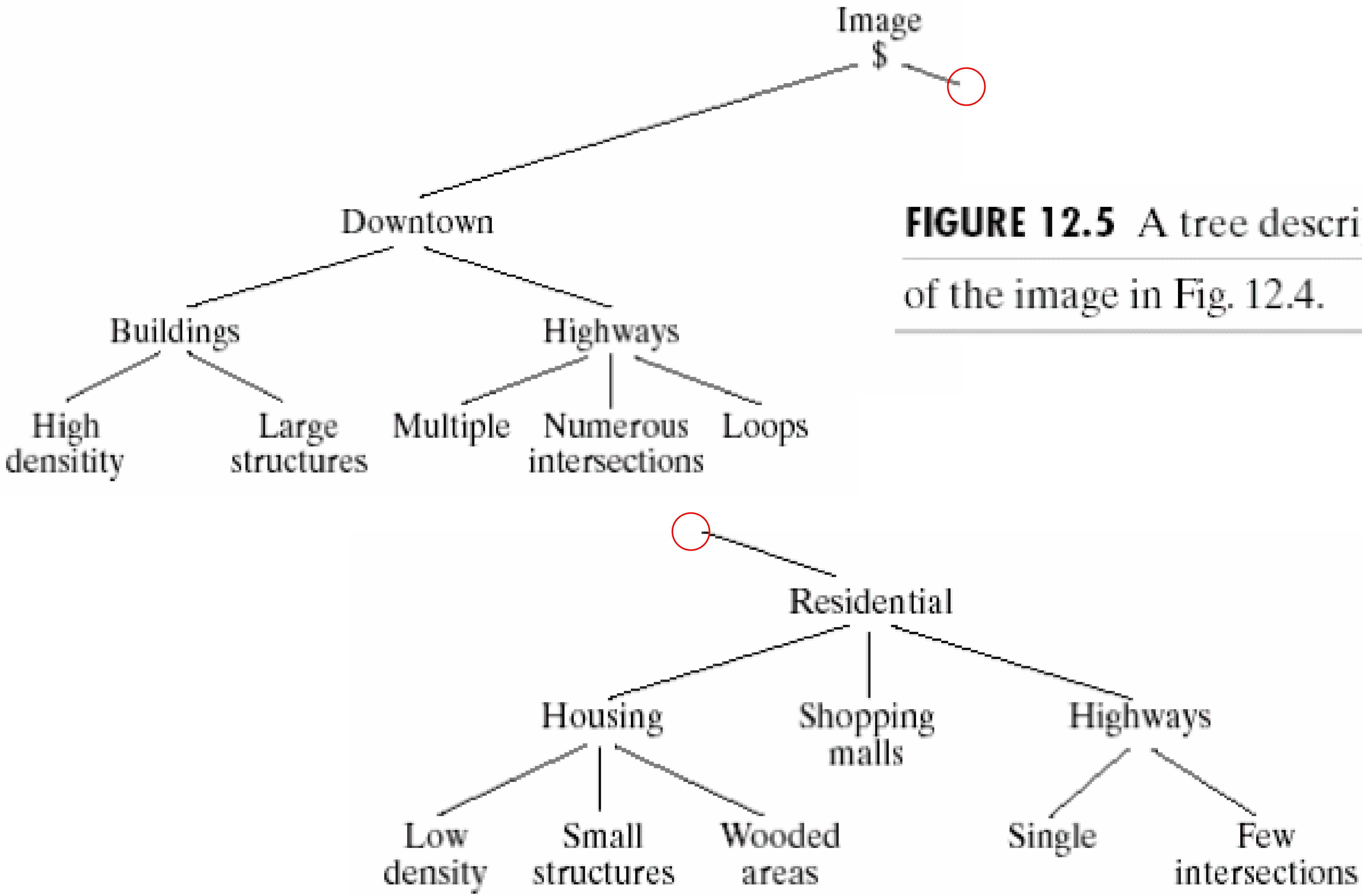


FIGURE 12.5 A tree description of the image in Fig. 12.4.

Object Recognition

- Example: 3 types of Iris are classified using their petal lengths and widths

virginica *versicolor* *setosa*

Object Recognition

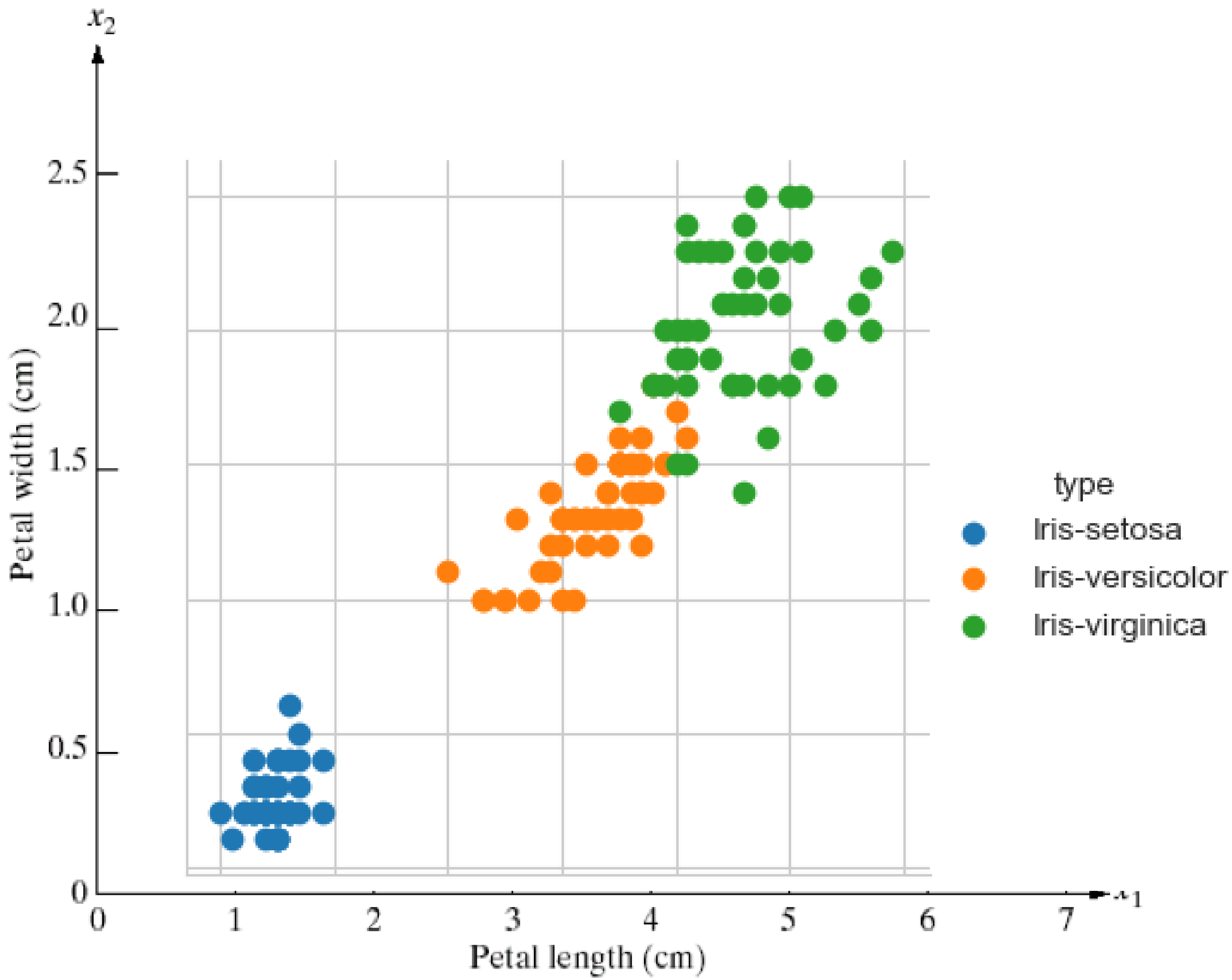
- We have three classes
  - *Iris virginica, Iris versicolor, Iris setosa* ω_1, ω_2 and ω_3
- Each flower is described using two features
  - Petal length, petal width

$$X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

- There are some differences of petal length and width between all classes
- There are also some variability within each class

Object Recognition

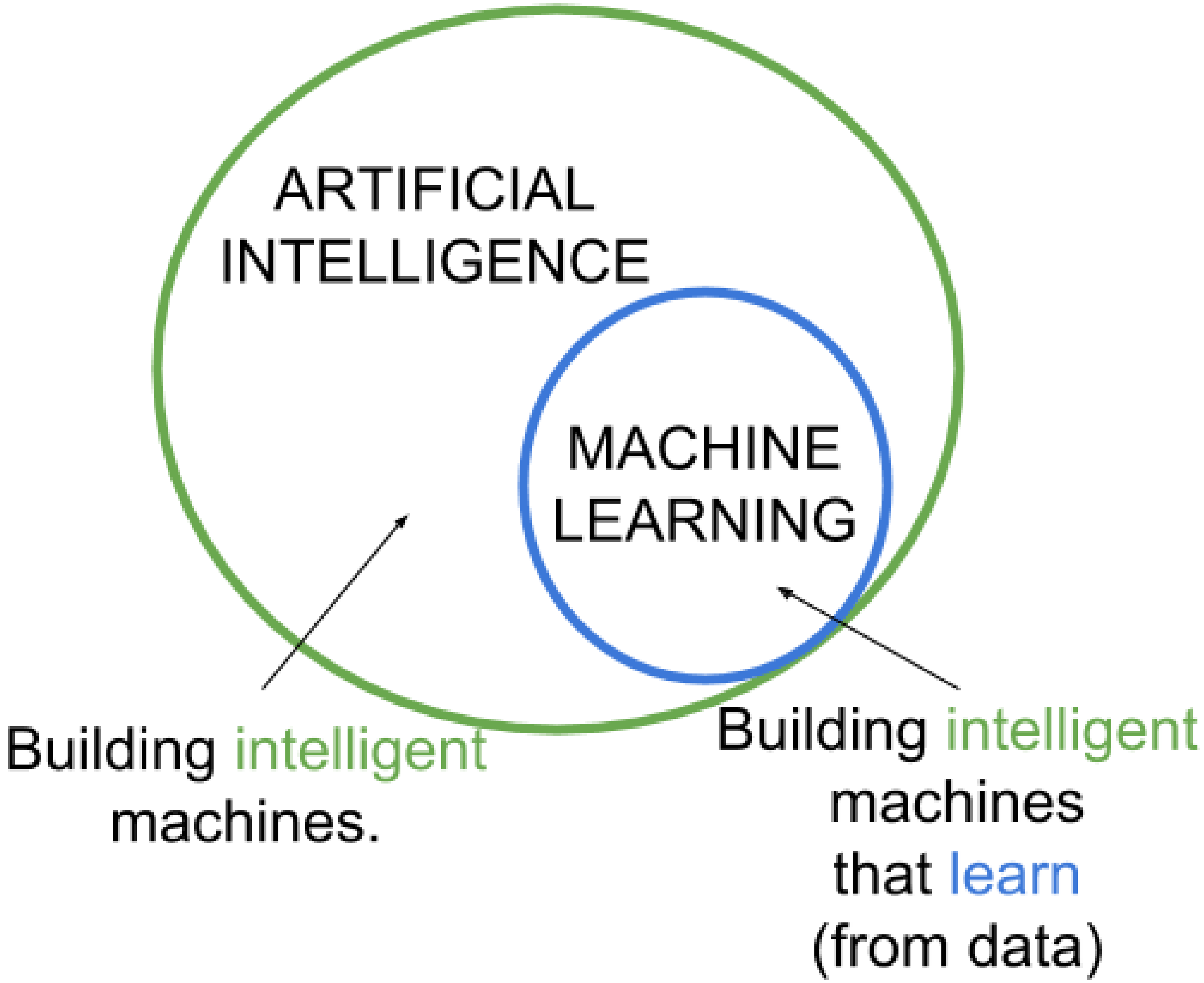
FIGURE 12.1
Three types of iris flowers described by two measurements.



Object Recognition

- The *setosa* type is well differentiate from the two others
- It is difficult to differentiate the two other types without error
- It is mainly a problem with the selection of good features

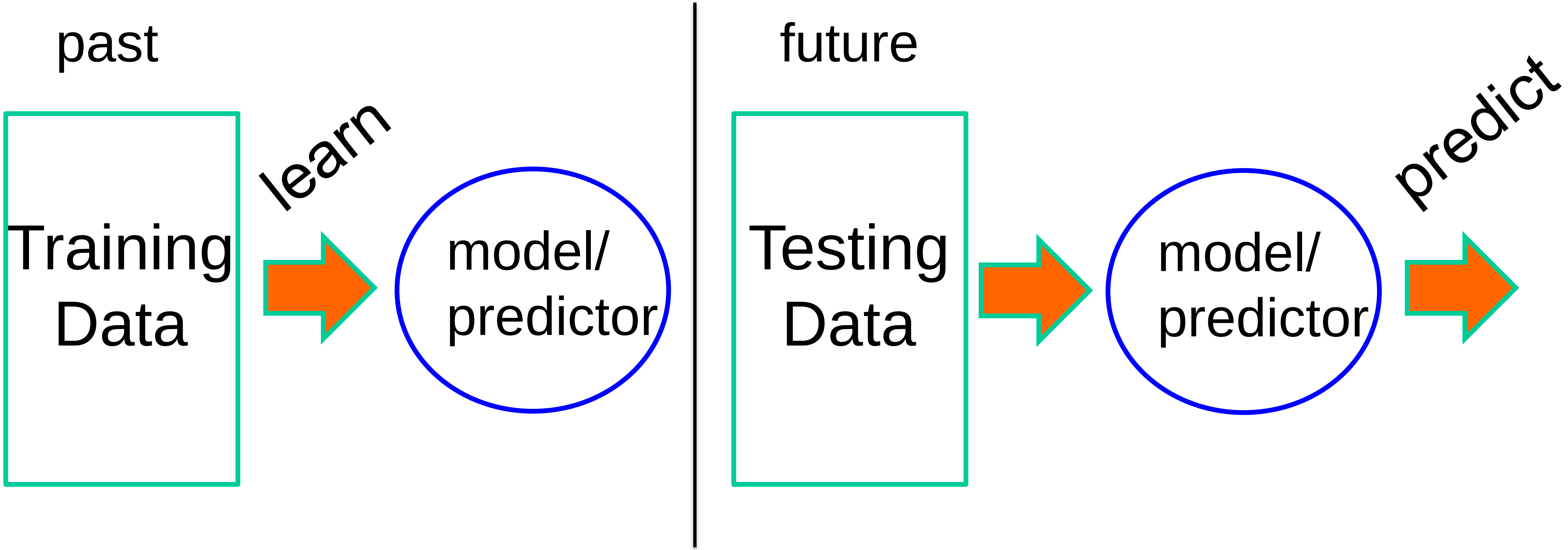
It is important to select discriminative features!


ARTIFICIAL INTELLIGENCE
MACHINE LEARNING
Building intelligent machines.
Building intelligent machines that learn (from data)

Machine Learning is…

Machine learning is about predicting the future based on the past.

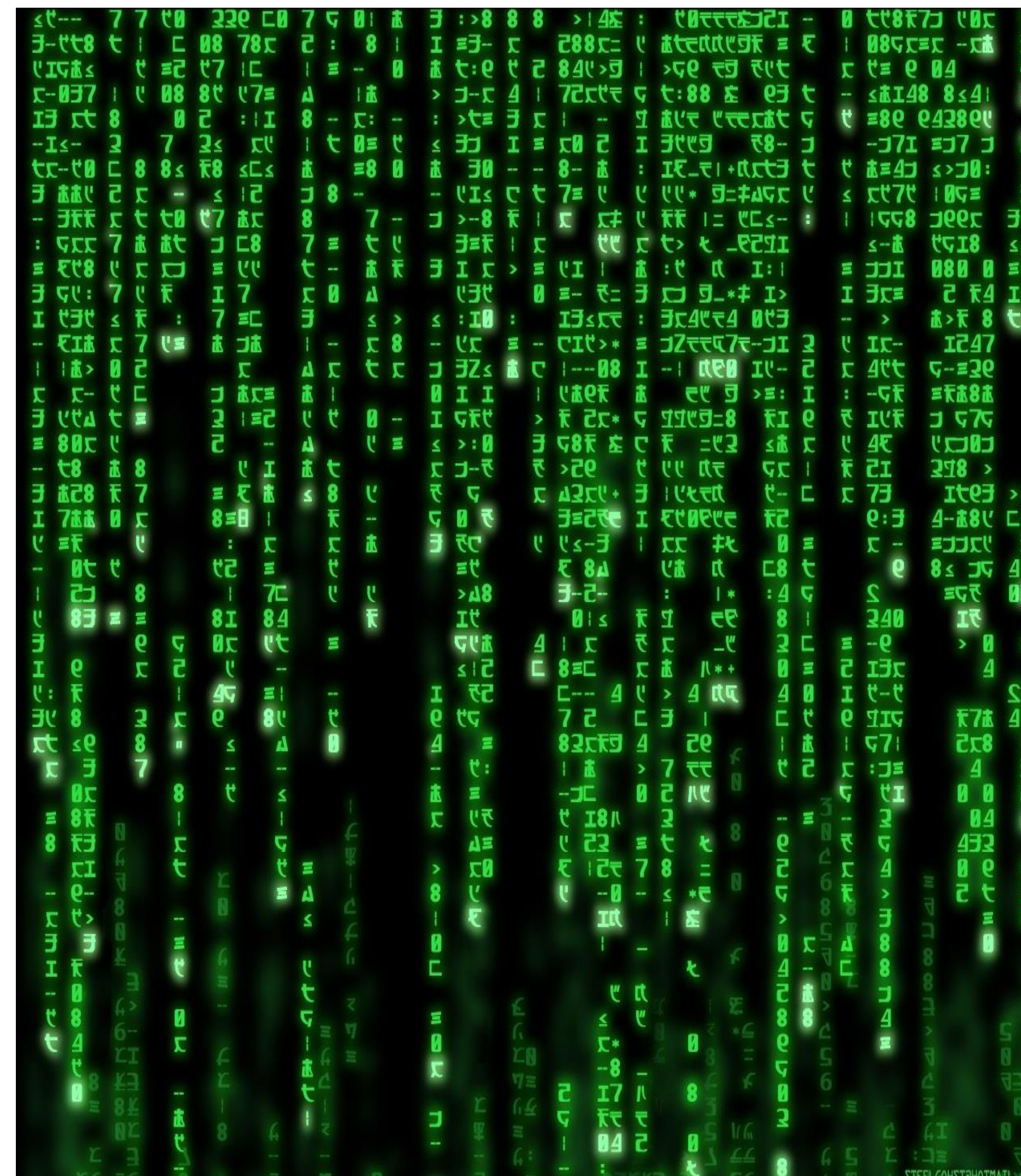
-- Hal Daume III



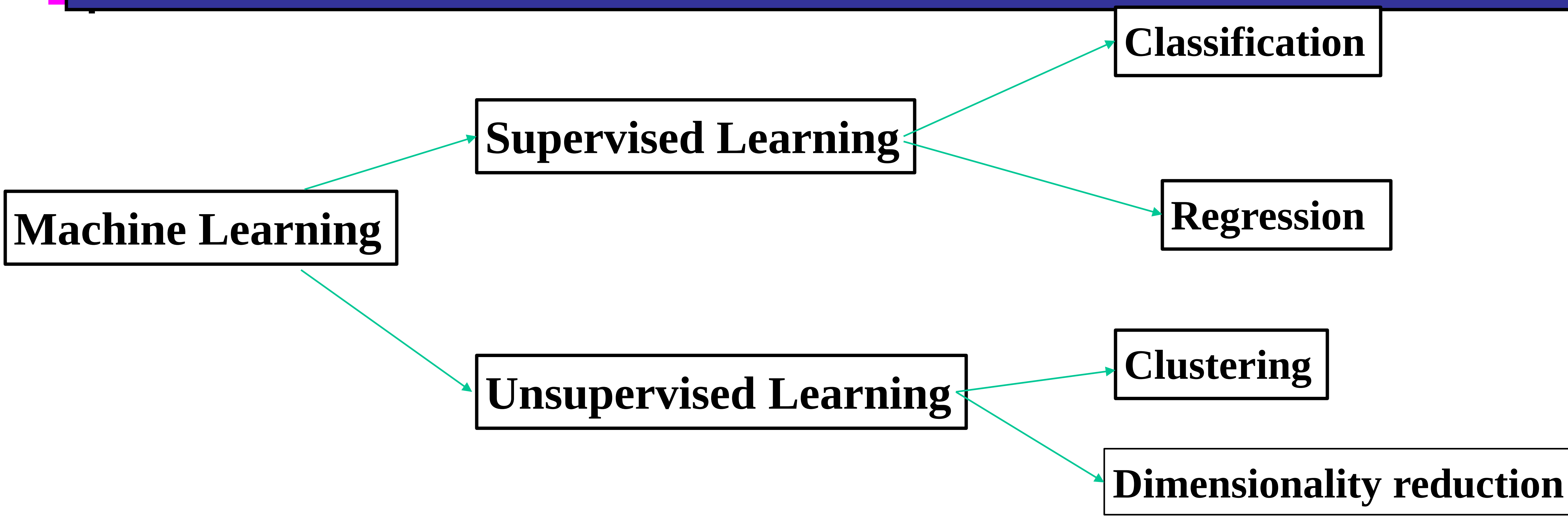
Why Machine Learning is Hard?

You See

Your ML Algorithm Sees
A Bunch of Bits



Types of Machine Learning Problems



- **Supervised Learning:** develop predictive models from labelled data (i.e. data with classes or targets)
- **Unsupervised learning:** describe hidden structure of unlabelled data
  - Clustering: Group similar data into categories (clusters) based only on input data
  - Dimensionality reduction: Reduce input variables of a dataset to a smaller set of variables (structure of dataset)

What does it mean to **<u>learn</u>** from data?

Momo Consumption

(in plates)

Thursday		
Breakfast	Lunch	Dinner
7	40	28

Momo Consumption

At a restaurant (in plates)

Thursday		
Breakfast	Lunch	Dinner
7	40	28

Momo Consumption

At a restaurant (in plates)

Thursday			Friday		
Breakfast	Lunch	Dinner	Breakfast	Lunch	Dinner
7	40	28	10	46	32

Momo Consumption

At a restaurant (in plates)

Thursday			Friday			Saturday		
Breakfast	Lunch	Dinner	Breakfast	Lunch	Dinner	Breakfast	Lunch	Dinner
7	40	28	10	46	32	8	43	

Momo Consumption

At a restaurant (in plates)

Thursday			Friday			Saturday		
Breakfast	Lunch	Dinner	Breakfast	Lunch	Dinner	Breakfast	Lunch	Dinner
7	40	28	10	46	32	8	43	

Patterns

Momo Consumption

At a restaurant (in plates)

Thursday			Friday			Saturday		
Breakfast	Lunch	Dinner	Breakfast	Lunch	Dinner	Breakfast	Lunch	Dinner
7	40	28	10	46	32	8	43	

Learning = "seeing" **Patterns**

Takeaway #1

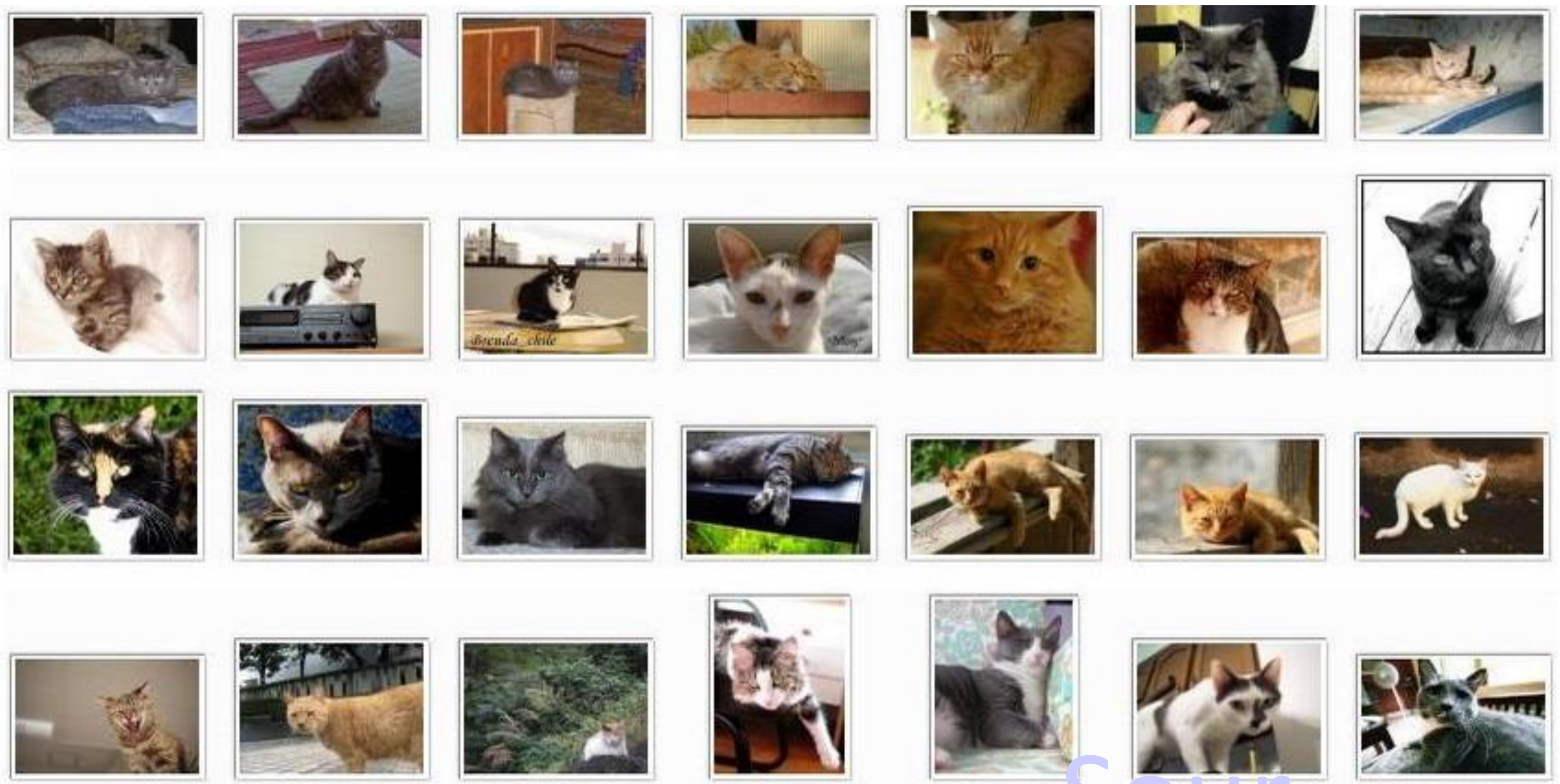
ML systems learn (gain knowledge, experience and pick up patterns) from data.

Why Machine Learning?

Why Machine Learning?

How can we detect cats?

Why Machine Learning?



Source

Why Machine Learning?
Because for a lot of problems we can’t explicitly define the solution.

Recognition Based on Decision Theoretic Methods

Let $x = (x_1, x_2, \ldots, x_n)^T$ for W pattern classes $\omega_1, \omega_2, \ldots, \omega_W$

$$d_i(x) > d_j(x) \quad j = 1, 2, \ldots, W; j \neq i$$

- In other words, an unknown pattern **x** is said to belong to the *i*th pattern class if, upon substitution of **x** into all decision functions, $d_i(x)$ yields the largest numerical value.

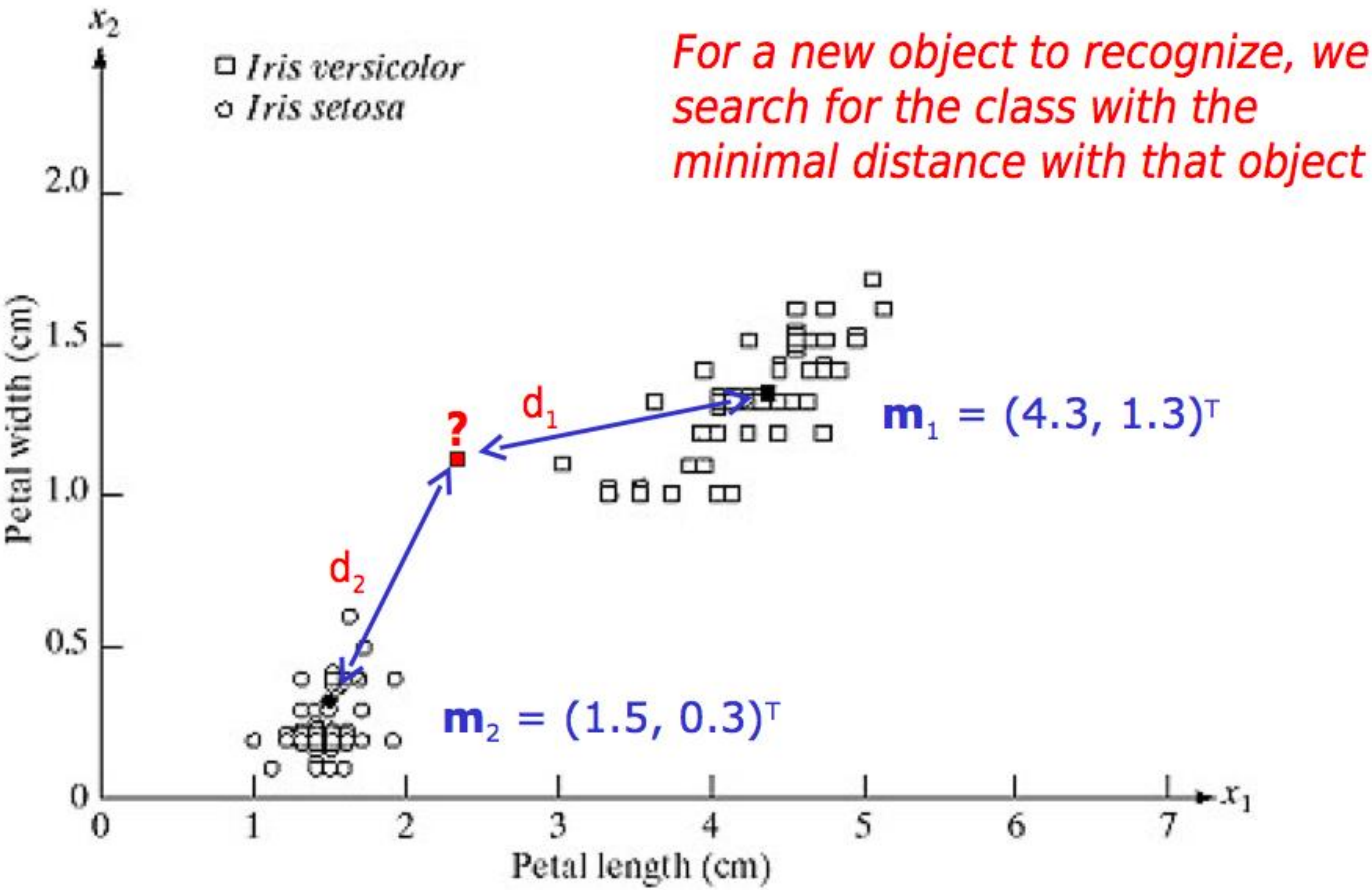
Minimum Distance Classifier

- Suppose that we define the prototype of each pattern class to be the mean vector of the patterns of that class: $m_j = \frac{1}{N_j} \sum_{x \in \omega_j} x_j \quad j = 1,2,\ldots,W$
- We then assign **x** to class ω_i if $D_i(\mathbf{x})$ is the smallest distance. $D_j(x) = \left\| x - m_j \right\|$

Minimum Distance Classifier

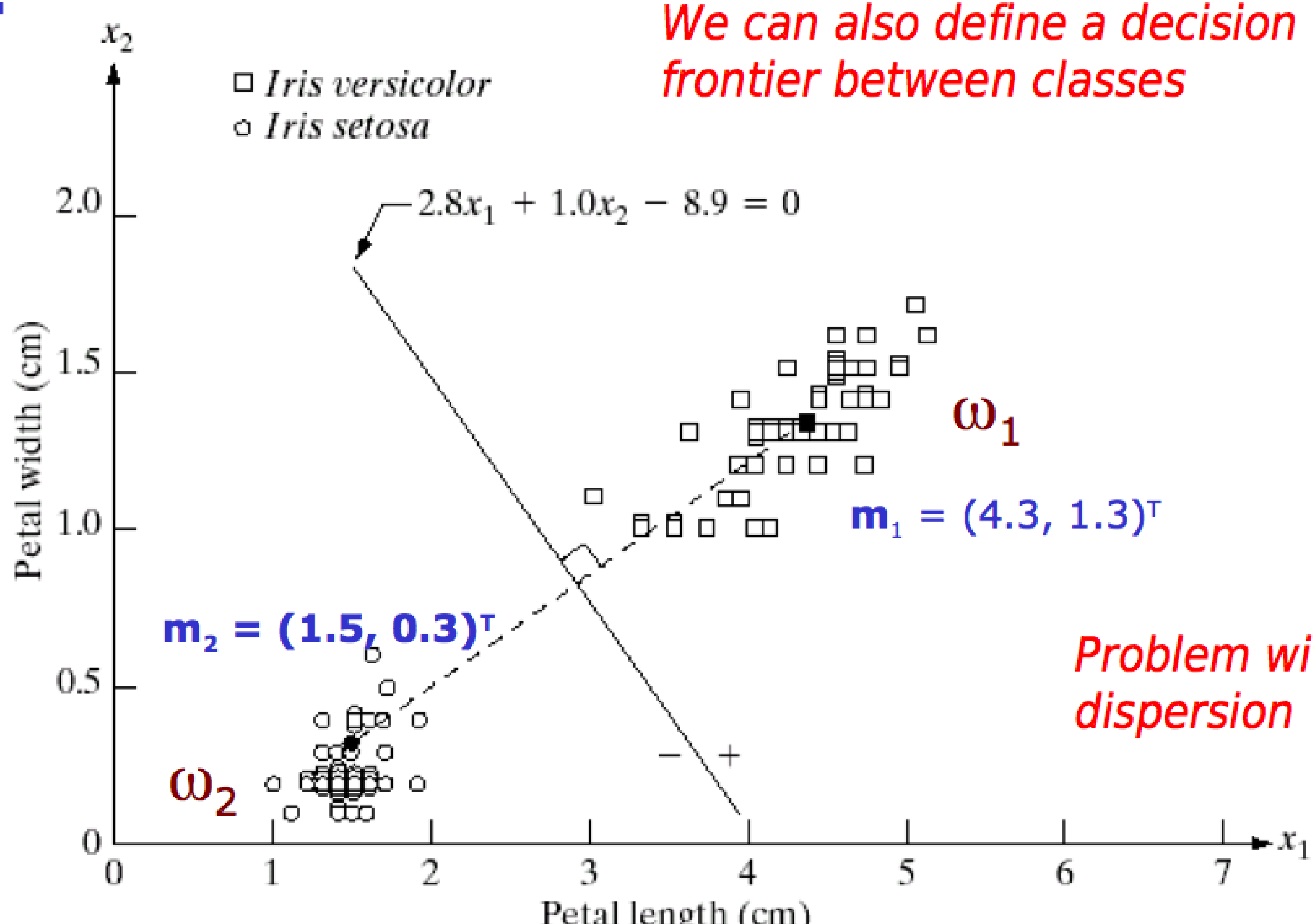
- It is not difficult to show (Problem 12.2) that selecting the smallest distance is equivalent to evaluating the functions $d_j(x) = x^T m_j - \frac{1}{2} m_j^T m_j \quad j = 1,2,\ldots,W$
- assign **x** to class ω_i if $d_i(\mathbf{x})$ is the largest numerical value.
- This formulation agrees with the concept of a decision function, as defined in Eq. (12.2-1).

Minimum Distance Classifier



Minimum Distance Classifier

We can also define a decision frontier between classes



Problem with axis dispersion

Matching By Correlation

- Correlation between a sub-image *w(x,y)* and an image *f(x,y)*
  - *w(x,y)* is of size J x K
  - *f(x,y)* is of size M x N
  - $J \le M$ and $K \le N$

- The correlation between *f(x,y)* and *w(x,y)* is:

$$c(x,y) = \sum_{s} \sum_{t} f(s,t)\, w(x+s, y+t)$$

Matching By Correlation

$$c(x,y)=\sum_s\sum_t f(s,t)w(x+s,y+t)$$

- *correlation coefficient*, which is defined as

$$\gamma(x,y)=\frac{\sum_s\sum_t\left[f(s,t)-\bar{f}(s,t)\right]\left[w(x+s,y+t)-\bar{w}\right]}{\left\{\sum_s\sum_t\left[f(s,t)-\bar{f}(s,t)\right]^2\sum_s\sum_t\left[w(x+s,y+t)-\bar{w}\right]^2\right\}^{\frac{1}{2}}}$$

$$r_{xy}=\frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}}$$

Matching By Correlation

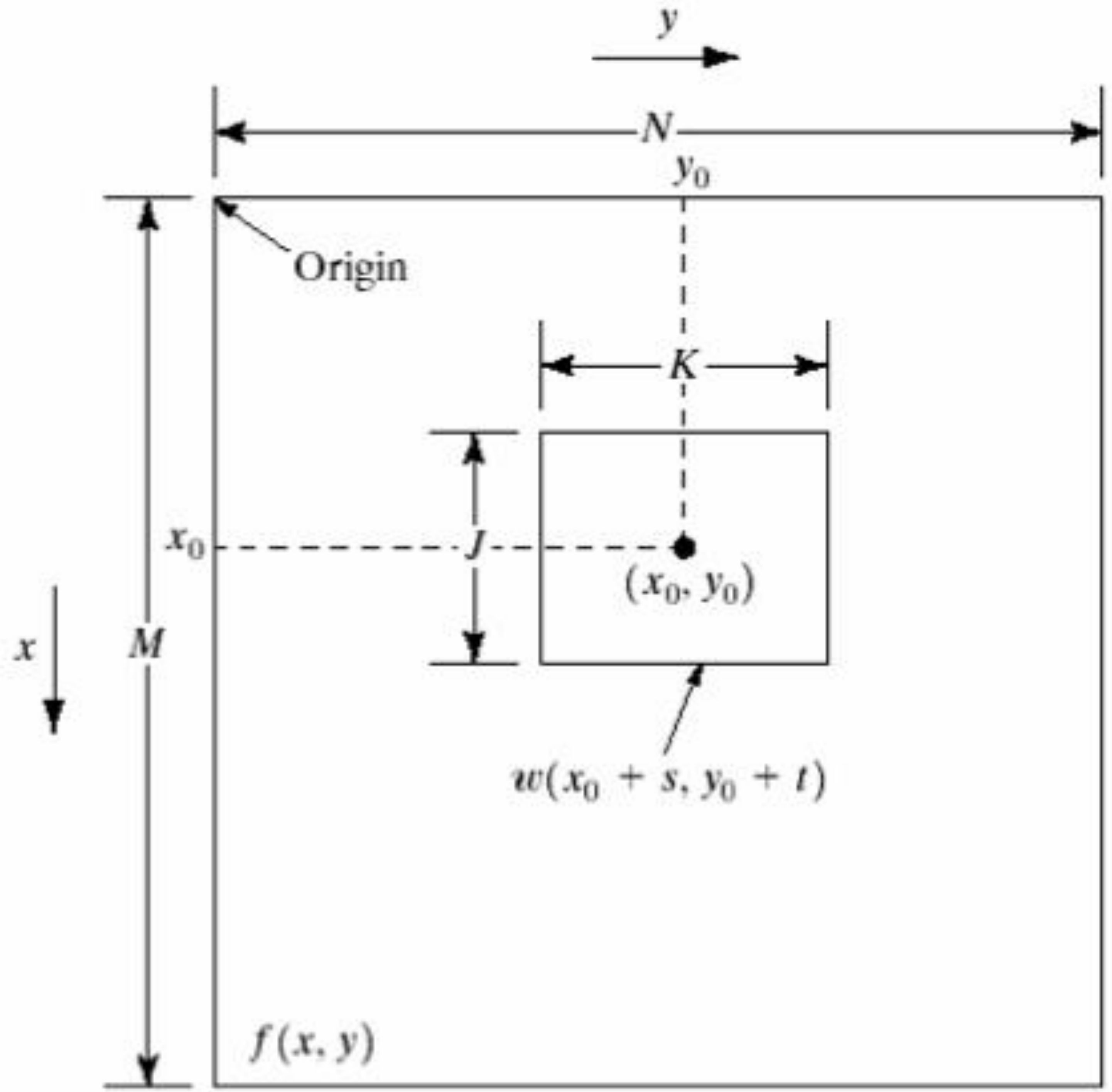
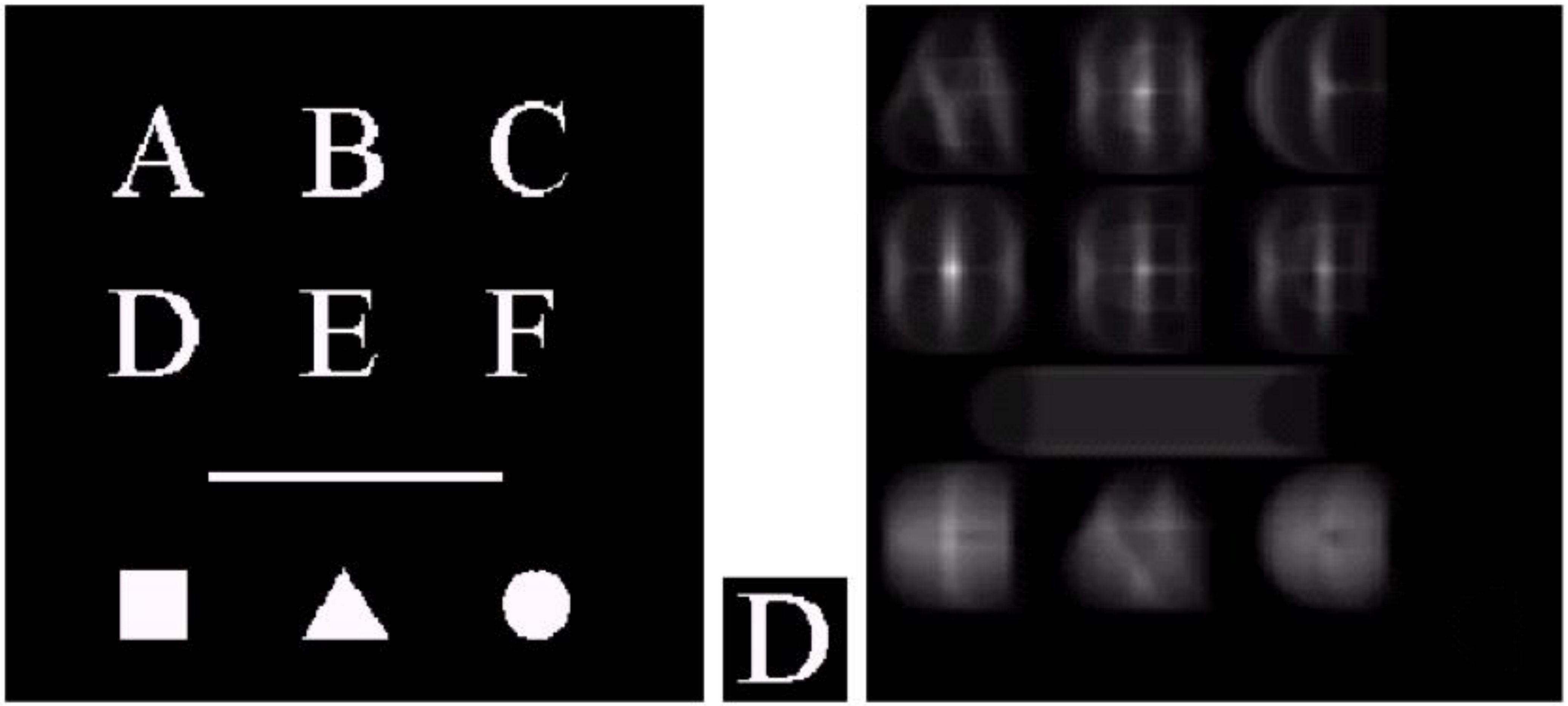


FIGURE 12.8 Arrangement for obtaining the correlation of f and w at point (x_0, y_0).

Matching By Correlation



a b c

FIGURE 12.9
(a) Image.
(b) Subimage.
(c) Correlation coefficient of (a) and (b). Note that the highest (brighter) point in (c) occurs when subimage (b) is coincident with the letter "D" in (a).

We look for the maximum correlation in the image

Neural Network

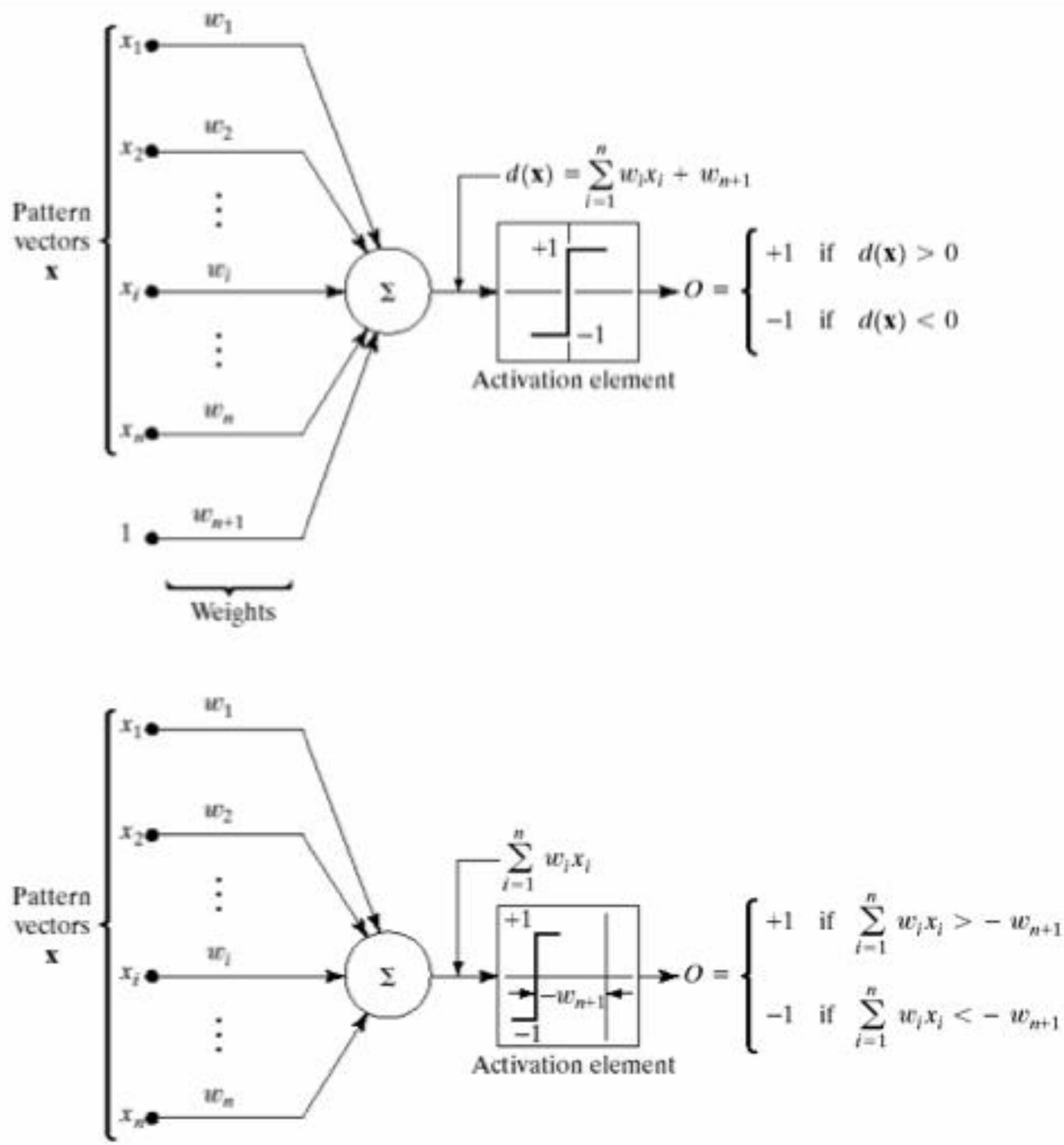


FIGURE 12.14 Two equivalent representations of the perceptron model for two pattern classes.

Neural Network : Training Algorithm

Linearly separable classes

, if $\mathbf{y}(k) \in \omega_1$ and $\mathbf{w}^T(k)\mathbf{y}(k) \le 0$, replace $\mathbf{w}(k)$ by

$$\mathbf{w}(k+1) = \mathbf{w}(k) + c\mathbf{y}(k) \qquad ($$

where c is a positive correction increment. Conversely, if $\mathbf{y}(k) \le \omega_2$ and $\mathbf{w}^T(k)\mathbf{y}(k) \ge 0$, replace $\mathbf{w}(k)$ with

$$\mathbf{w}(k+1) = \mathbf{w}(k) - c\mathbf{y}(k). \qquad (12.2\text{-}35)$$

Otherwise, leave $\mathbf{w}(k)$ unchanged:

$$\mathbf{w}(k+1) = \mathbf{w}(k).$$

Neural Network : Training Algorithm

yielding the training set $\{(0,0,1)^T, (0,1,1)^T\}$ for class ω_1 and $\{(1,0,1)^T, (1,1,1)^T\}$ for class ω_2. Letting $c = 1, \mathbf{w}(1) = \mathbf{0}$, and presenting the patterns in order results in the following sequence of steps:

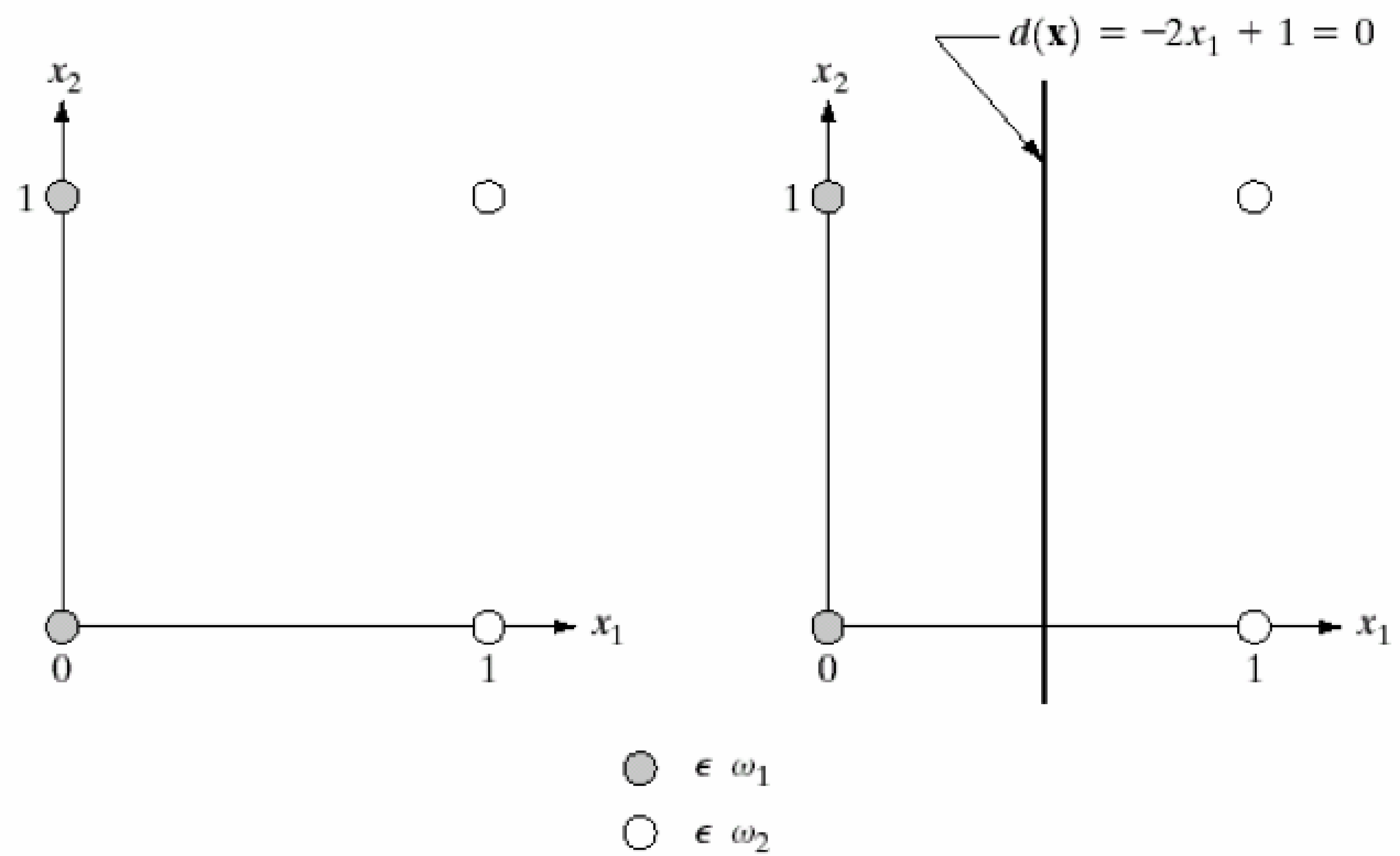
$$\mathbf{w}^T(1)\mathbf{y}(1) = [0, 0, 0]\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = 0 \qquad \mathbf{w}(2) = \mathbf{w}(1) + \mathbf{y}(1) = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$\mathbf{w}^T(2)\mathbf{y}(2) = [0, 0, 1]\begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} = 1 \qquad \mathbf{w}(3) = \mathbf{w}(2) = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$\mathbf{w}^T(3)\mathbf{y}(3) = [0, 0, 1]\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = 1 \qquad \mathbf{w}(4) = \mathbf{w}(3) - \mathbf{y}(3) = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{w}^T(4)\mathbf{y}(4) = [-1, 0, 0]\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = -1 \qquad \mathbf{w}(5) = \mathbf{w}(4) = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix}$$

Neural Network : Training Algorithm



a b

FIGURE 12.15
(a) Patterns belonging to two classes.
(b) Decision boundary determined by training.

Neural Network : Training Algorithm

Nonseparable classes

Consider the criterion function

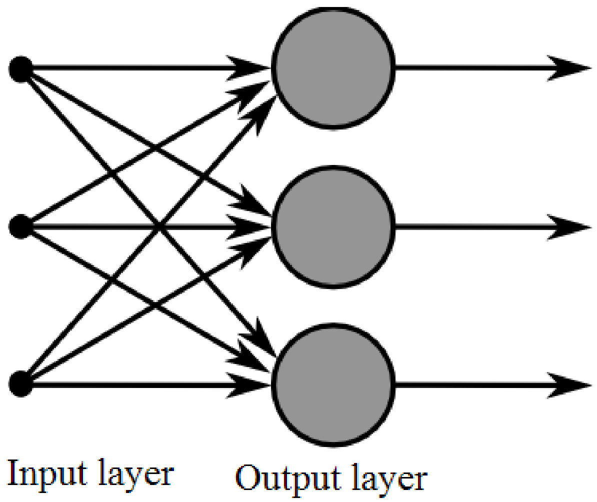
$$J(\mathbf{w}) = \frac{1}{2}(r - \mathbf{w}^T\mathbf{y})^2$$

$$w(k+1) = w(k) - \alpha\left[\frac{\partial J(w)}{\partial w}\right]_{w=w(k)}$$

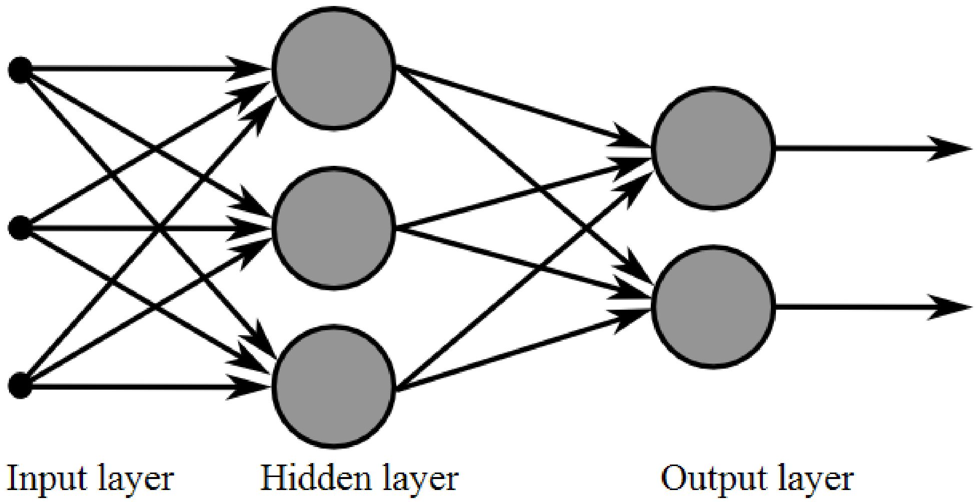
$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = -(r - \mathbf{w}^T\mathbf{y})\mathbf{y}.$$

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \alpha\left[r(k) - \mathbf{w}^T(k)\mathbf{y}(k)\right]\mathbf{y}(k)$$

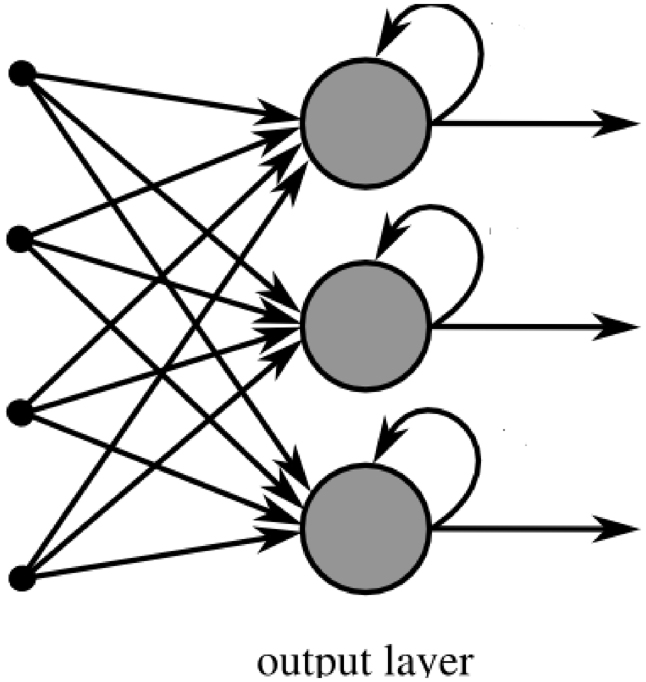
Examples of ANN topologies



Single layer ANN



Multilayer ANN



ANN with one recurrent layer

Fundamentals of learning and training samples

- **The weights in a neural network are the most important factor in determining its function.**
- **A training set is a set of training patterns, which we use to train our neural net.**
- **Training is the act of presenting the network with some sample data and modifying the weights to better approximate the desired function**

Neural Network : Training Algorithm

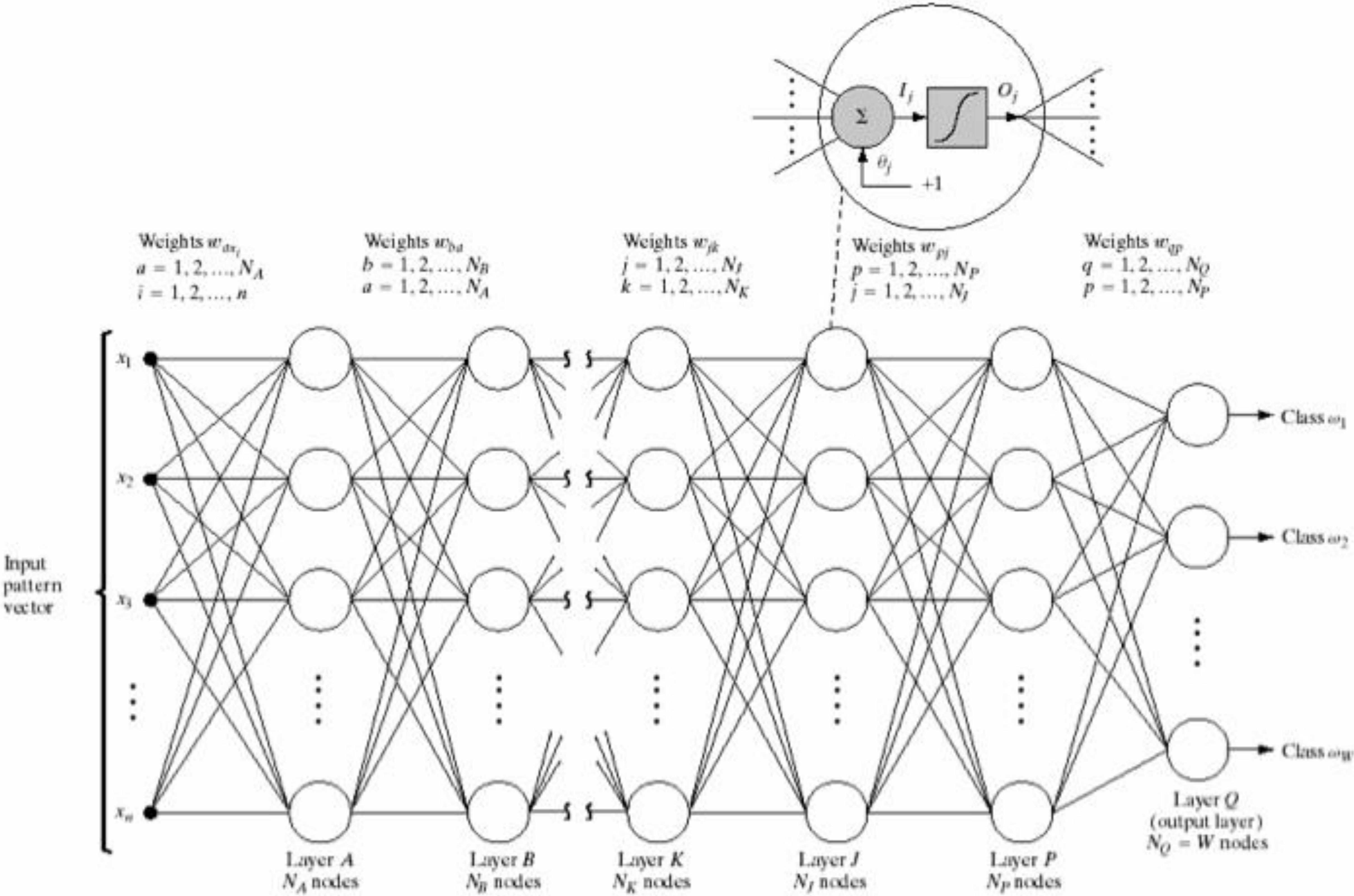


FIGURE 12.16 Multilayer feedforward neural network model. The blowup shows the basic structure of each neuron element throughout the network. The offset, θ_j, is treated as just another weight.

Neural Network : Training Algorithm

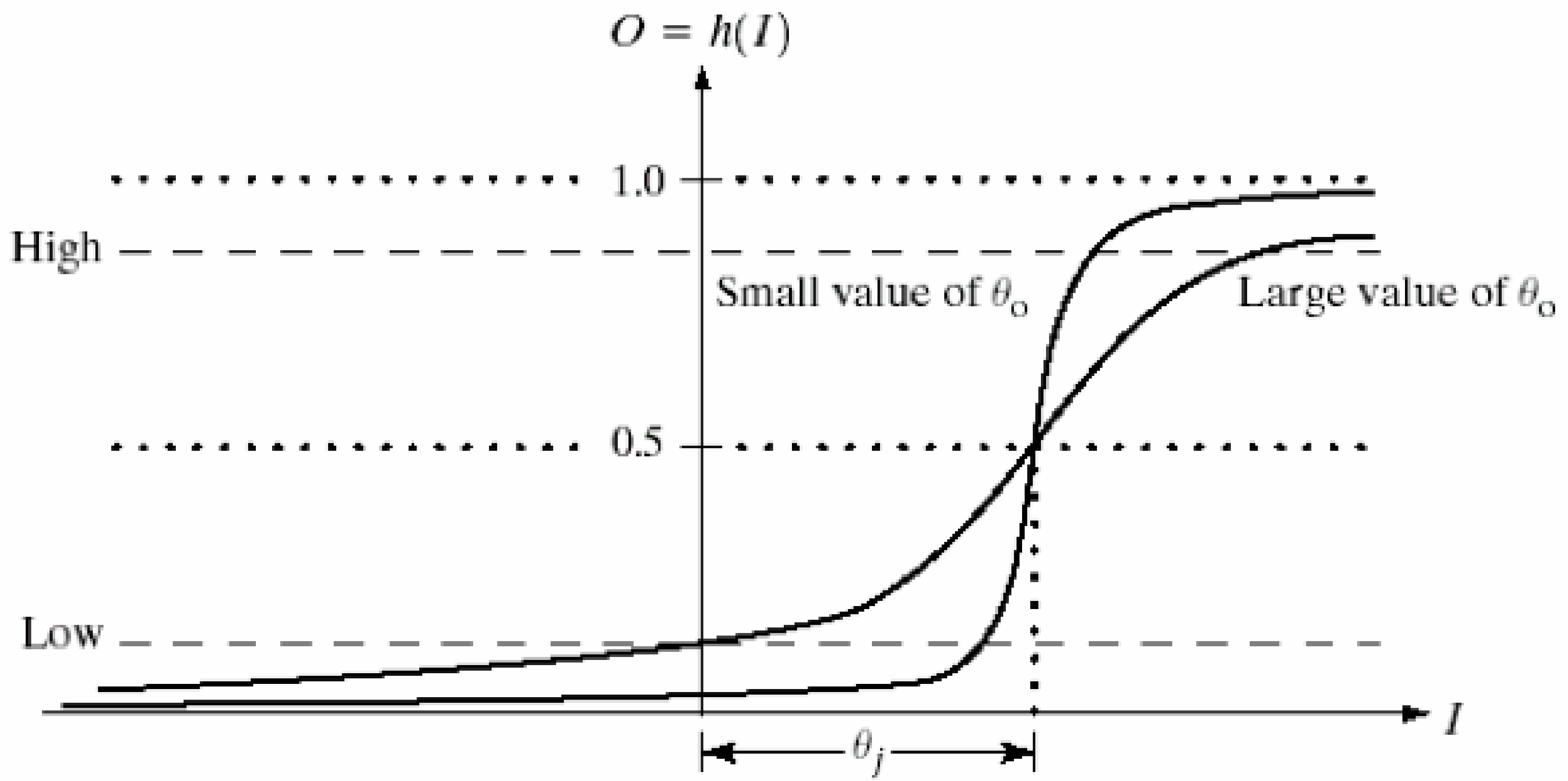


FIGURE 12.17 The sigmoidal activation function of Eq. (12.2-47).

http://en.wikipedia.org/wiki/Hopfield_network

http://home.agh.edu.pl/~vlsi/AI/hamming_en/

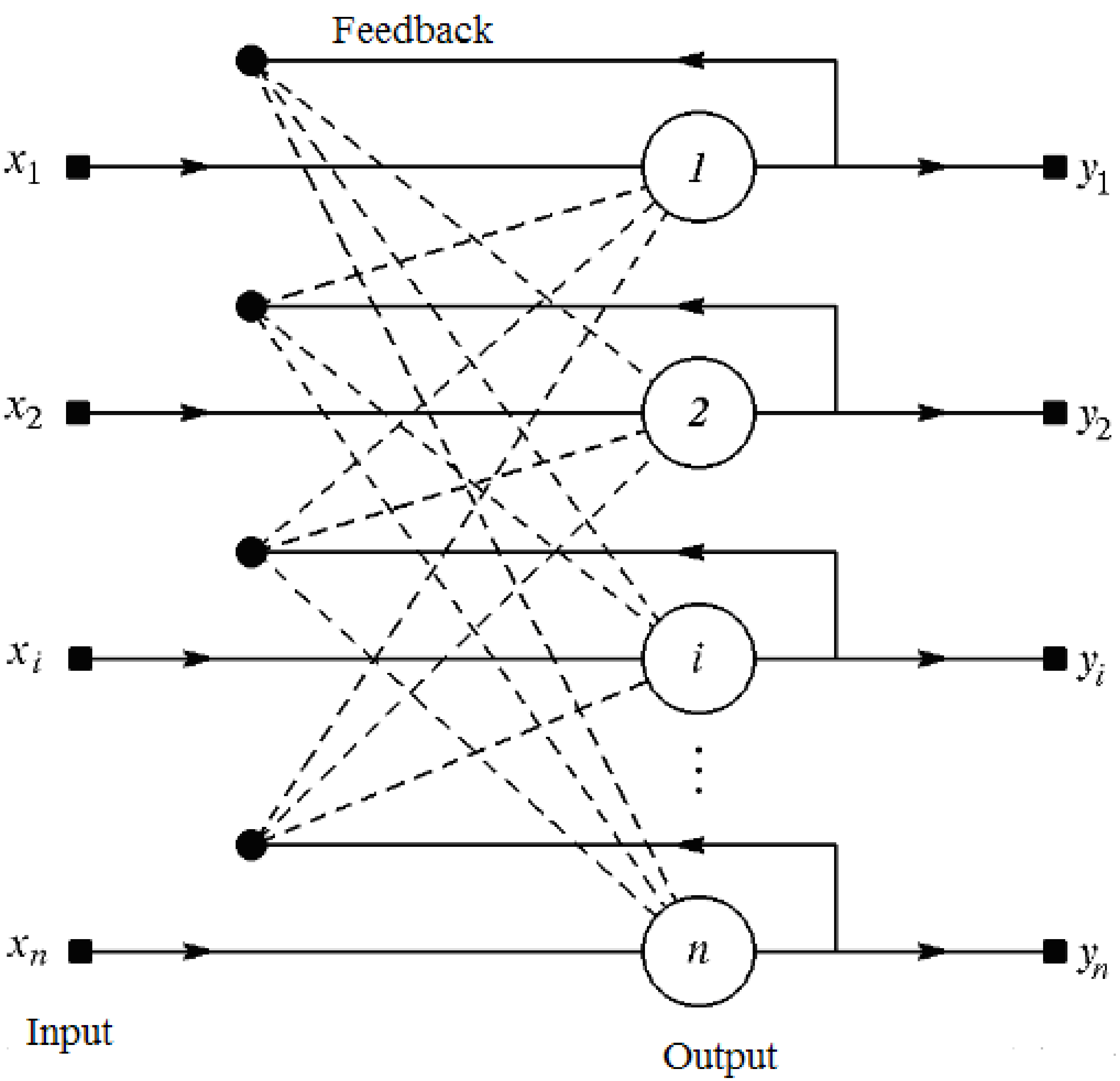
Using neural networks in practice (discussion)

- **Classification**
  - in marketing: consumer spending pattern classification
  - In defence: radar and sonar image classification
  - In medicine: ultrasound and electrocardiogram image classification, EEGs, medical diagnosis
- **Recognition and identification**
  - In general computing and telecommunications: speech, vision and handwriting recognition
  - In finance: signature verification and bank note verification
- **Assessment**
  - In engineering: product inspection monitoring and control
  - In defence: target tracking
  - In security: motion detection, surveillance image analysis and fingerprint matching
- **Forecasting and prediction**
  - In finance: foreign exchange rate and stock market forecasting
  - In agriculture: crop yield forecasting
  - In marketing: sales forecasting
  - In meteorology: weather prediction

Recurrent neural networks

- Capable to influence to themselves by means of recurrences, e.g. by including the network output in the following computation steps.
- Hopfield neural network
- Hamming neural network

Hopfield Neural Network



- A Hopfield network is a form of recurrent artificial neural network proposed by John Hopfield in 1982
- Hopfield networks serve as content-addressable ("associative") memory systems with binary threshold nodes, or with continuous variables. (-1,1 or 0,1)
- Hopfield networks also provide a model for understanding human memory.

Hopfield Neural Networks

- Effectively, Hopfield Neural Networks are like **associative memory**:

- Wish to store **p** patterns ς^{μ}_{i} so that *given* a new input pattern ς_i, the network
  responds by producing the stored pattern most closely resembling ς_i.

- Stored patterns are indexed by superscript μ**=1, ,p** while the nodes in the neural network are labeled **i = 1, , n**.

- Since knowledge is encoded in the network during design and not learnt, this type of network uses **unsupervised learning**.

Example (1 of 4)

- Consider designing a neural network to classify the hand written digits 0 through 9 that are presented on a 16x16 grid of pixels.

How would....

- A feed-forward Neural Network using back-propagation be designed and operate?

- A Hopfield Neural Network be designed and operate?

Example (2 of 4) - A Back-Propagatio

- Back-propagation Neural Network might be designed and operate as follows:

- Consist of 16x16 input nodes. (256, one for each pixel)
- Consist of 10 output nodes. (one for each digit to recognize)
- Consist of some number of hidden nodes.
- Require training using **sample digits** to adapt/adjust the weights (**learning**).
- Process new digits after training.

- Given a set of 16x16 input pixels, 1 of 10 outputs would become active, indicating the classification of the input digit.

- This is classification using a **functional representation** (i.e., there is a non-linear function encoded in the network mapping 16x16 inputs to 10 outputs).

Example (3 of 4) - A Hopfield Net

A Hopfield Neural Network might be designed and operate as follows:

- A 16x16 array of nodes is created.
- The array of nodes is fully connected (edges between all pairs of nodes).
- **Each node is an input and an output. (one layer of neurons only!)**
- Edge weights are determined **a priori based on ideal patterns** for digits 0 through 9; patterns are stored in the network.
  - Continued, next page…

Example (4 of 4) - A Hopfield Net

- Given an unknown input, we initialize the output of each node (i.e., the output at time t=0).

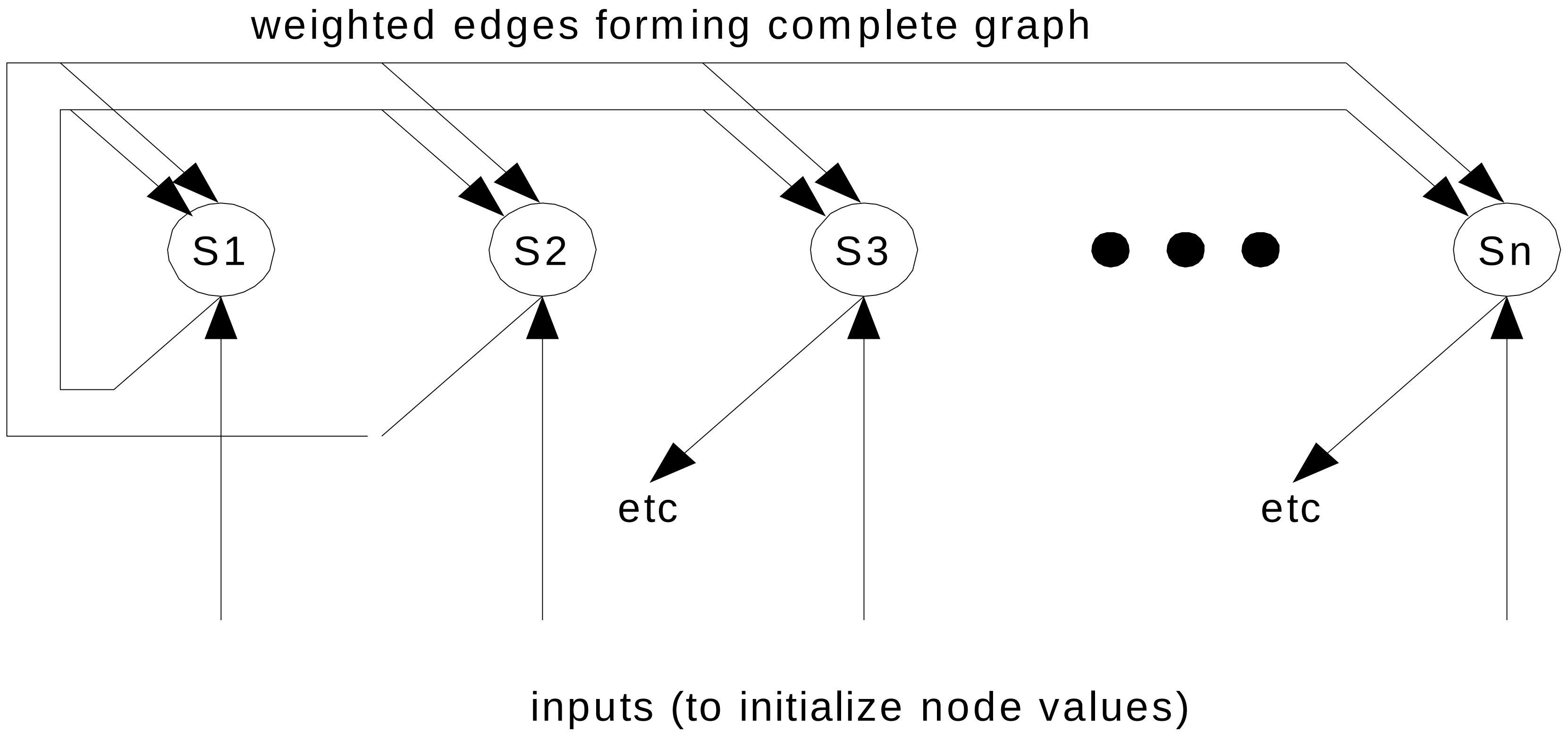
- The network is allowed to **evolve over time until the node outputs become stable**.
  - Will not always stabilize.

- The final output values will (hopefully) correspond to one of the memorized patterns of ideal digits we stored in our network.
  - When might it not? more later …

- This is classification using an **associative representation** in that the network tries to find the **memorized pattern** most closely representing the input.

Hopfield Neural Network Topology

- Basically, a complete weighted graph:



- Each node has an activation function (assume sign function so outputs are -1 or +1):

$$S_i(t) = sign\left(\sum_{j=1}^{N} w_{ij} S_j(t-1)\right)$$

- **Note**: outputs are a **function of time**. The value at time t depends on values at time (t-1).

Algorithm (for “running” the network)

1. Assume network weights are determined and input pattern ζ given.
2. Set $S_i(0) = \zeta_i$ /* initialize network. */
3. Set $t = 1$.
4. Compute $S_i(t) = sign(\sum_{j=1}^{N} w_{ij} S_j(t-1))$.
5. If $S_i(t) == S_i(t{-}1)$ then STOP and GOTO step 6; otherwise $t = t{+}1$ and GOTO step 4.
6. The pattern most resembling the input is now available (as the output of the nodes)

Storing Single Patterns (1 of 2) (Stability)

- Storing a pattern is equivalent to asking "how do we pick network weights?"

- Assume we wish to store a pattern:

  ζ with bits $\zeta_i = \pm 1 \quad \forall i = 1, \cdots, N$

- **Observation: If we were to present the stored pattern as input to the network (i.e., hit it exactly), the outputs of the network should not change!**

  - Since the input equals the desired output!

- We have a stability condition given by:

$$\zeta_i = sign(\sum_{j=1}^{N} w_{ij}\zeta_j) \text{ since } S_i = \zeta_i \text{ for all time}$$

Storing Single Patterns (2 of 2) (Weight Selection)

- Consider the following selection for weights:

$$w_{ij} = \frac{\zeta_i \zeta_j}{N}$$

- If we substitute these weights into the stability equation we get:

$$sign(\sum_{j=1}^{N} w_{ij}\zeta_j) = sign(\sum_{j=1}^{N} \frac{\zeta_i\zeta_j}{N}\zeta_j) = sign(\zeta_i \sum_{j=1}^{N} \frac{\zeta_j^2}{N}) = sign(\zeta_i) = \zeta_i$$

- So, the output will not change if the input pattern is exact.

Storing Multiple Patterns

- How can we store p patterns? i.e., we have patterns

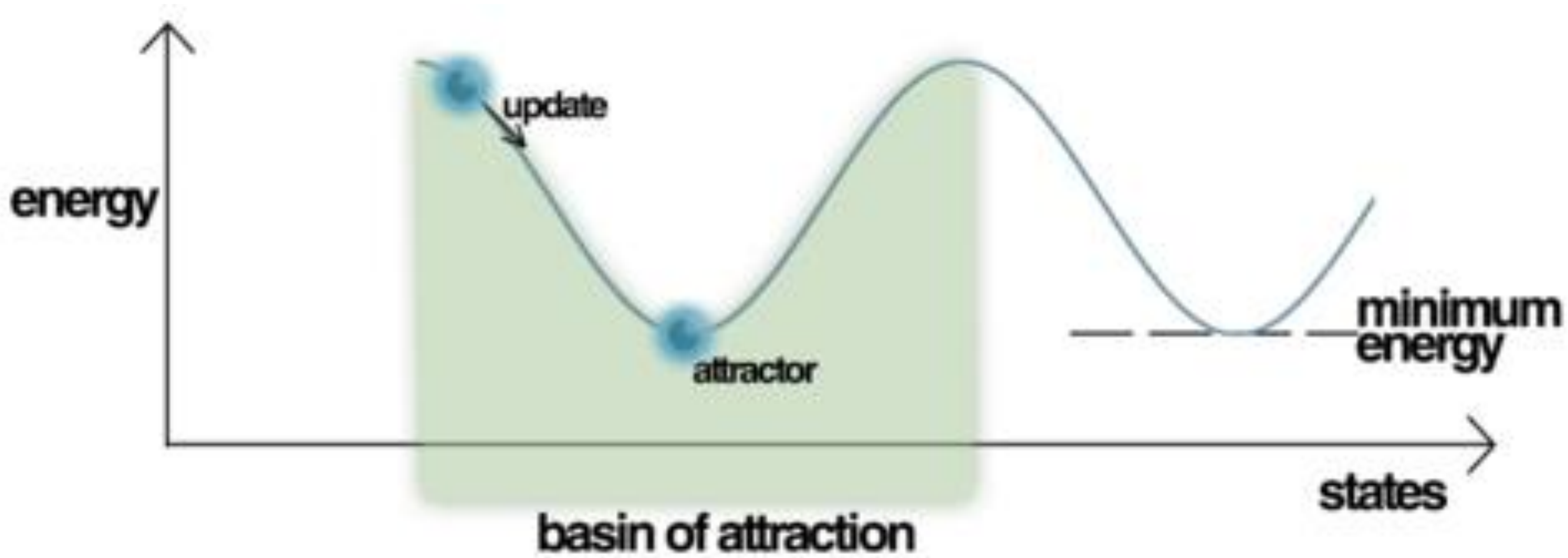
$$\zeta_1, \zeta_2, \cdots, \zeta_p$$

- Solution: pick weights as a superposition of the stored patterns:

$$w_{ij} = \frac{1}{N} \sum_{k=1}^{p} \zeta_i^k \zeta_j^k$$

- In other words, compute the ideal weight for each pattern stored individually, and then average them out to pick the final, single weight for each edge.

Hopfield Neural Network: Energy Functions

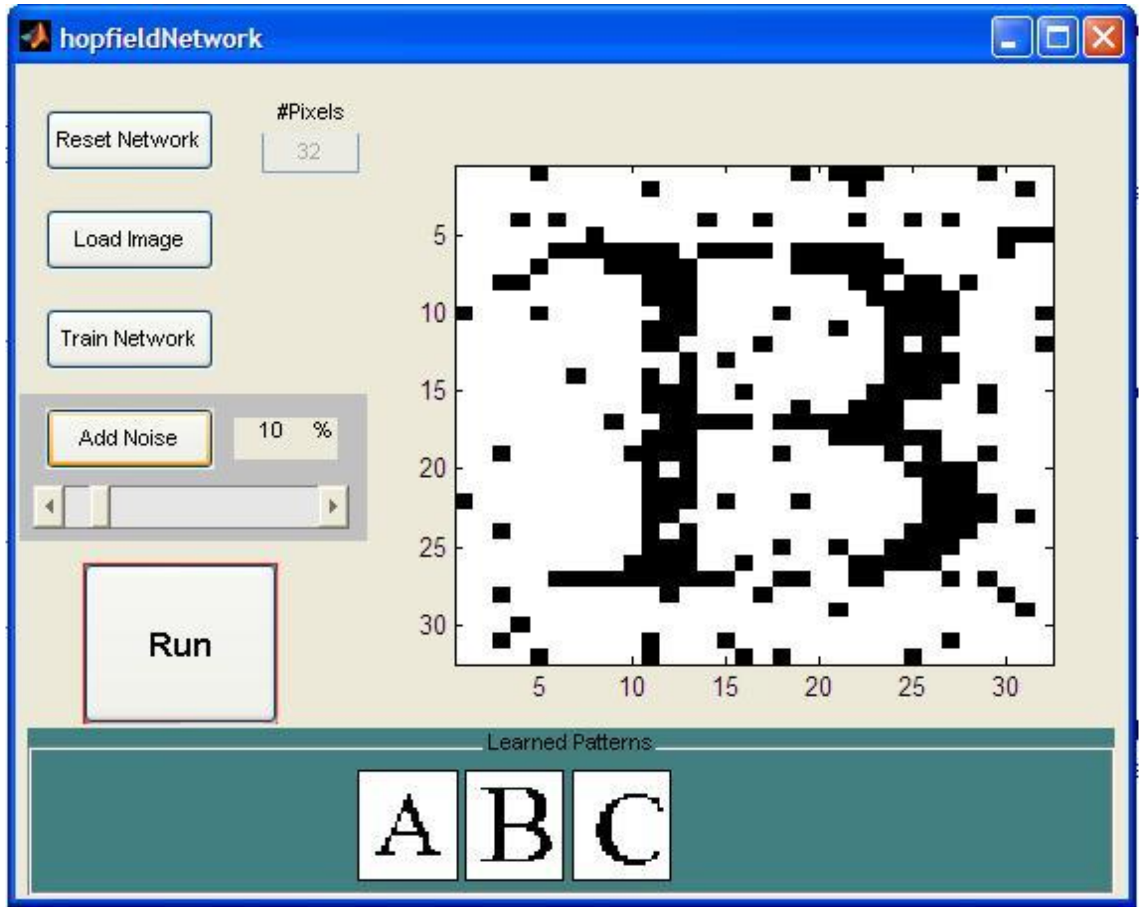


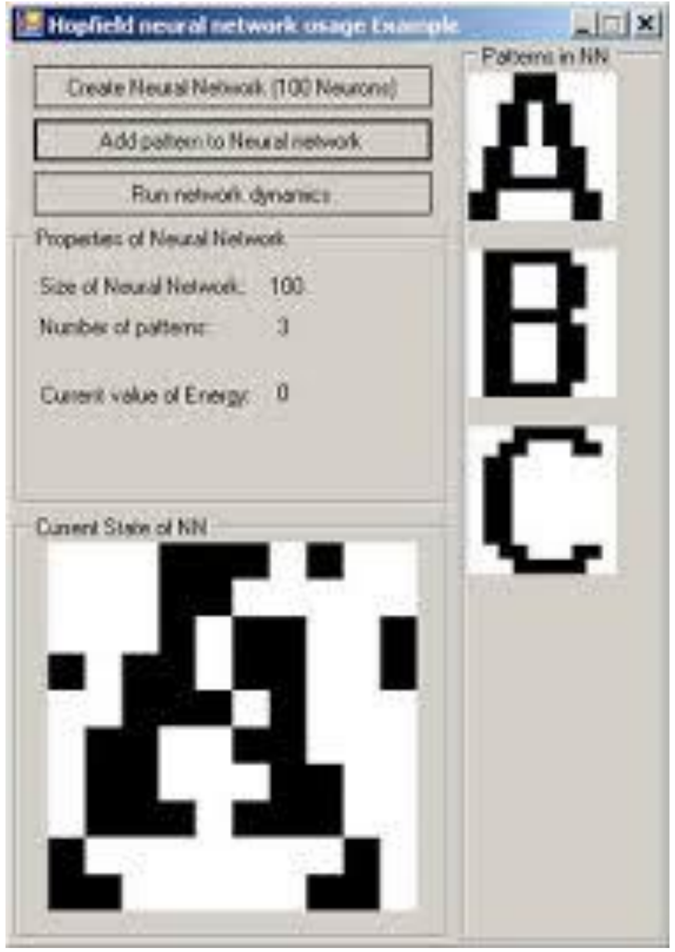
$$H = -\frac{1}{2} \sum_{i,j=1}^{N} w_{ij} S_i S_j$$

- There is a relationship between dynamics (convergence to a stored piece of information) of Hopfield Network and an **Energy Function.**
- Consider that our weights are picked such that the stored patterns represent local minimums of the function **H.**
- Matching an input to a stored pattern is like minimization of H; “fall into the closest minimum”.
- The stability condition is akin to being stuck in the local minimum.

Hopfield Neural Network

associative memory



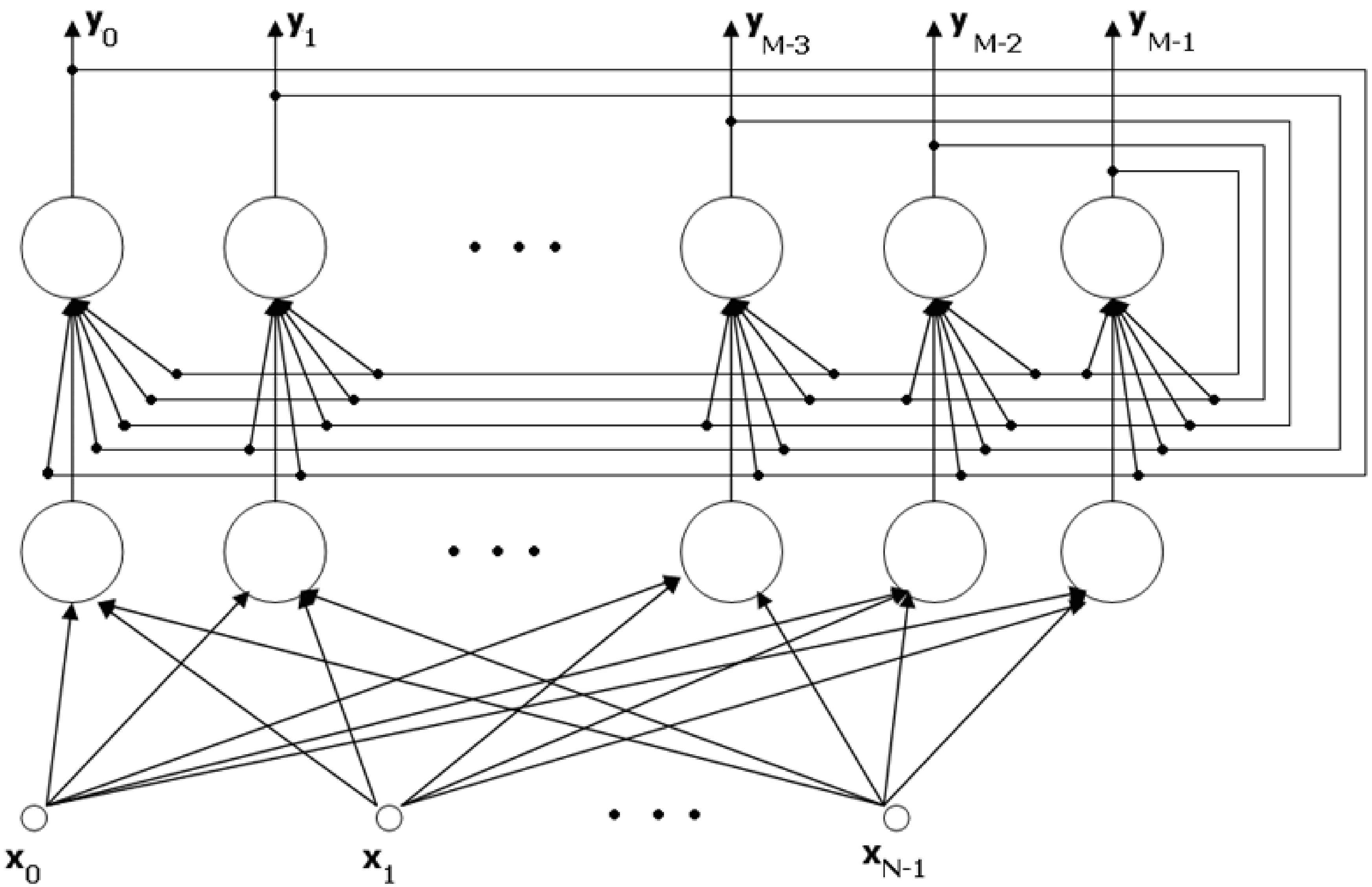


Limitations of Hopfield Neural Network

- Severely limited in the number of patterns p that can be stored reliably in N nodes.
- Generally, the maximum number of patterns must be below 0.15N for reliable performance;
  - Avalanche in error occurs above 0.138N.
- Too many patterns will result in spurious outputs; i.e., outputs not corresponding to any stored pattern.

- Storing similar patterns can cause errors in output.

Hamming Neural Network

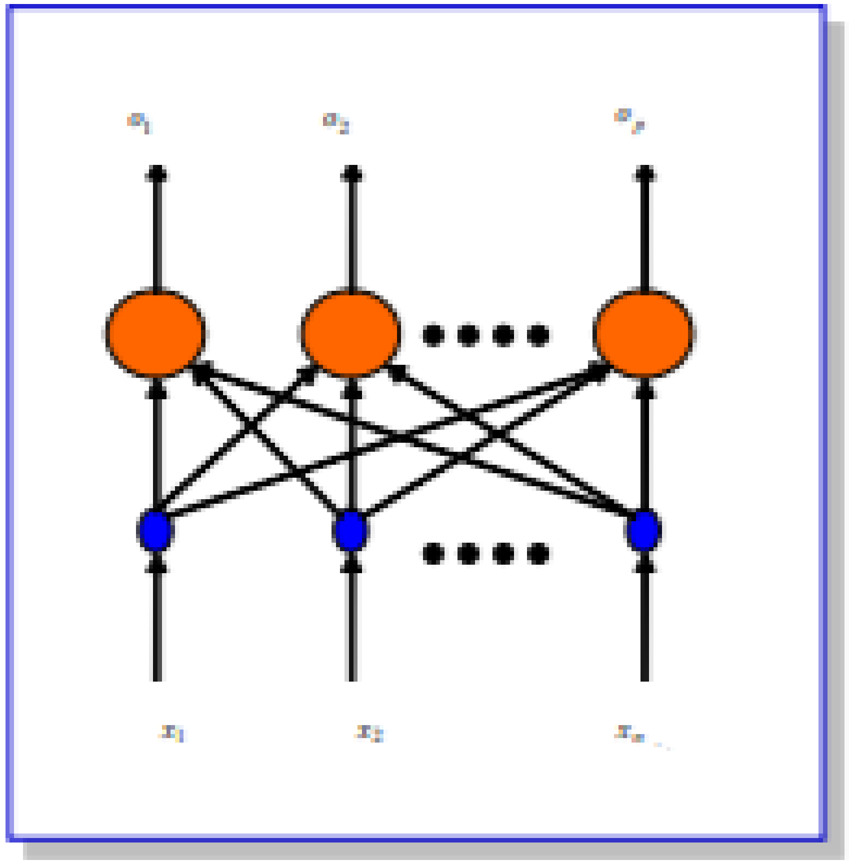


- The Hamming network performs the task of pattern association, or classification, based on measuring the Hamming distance

- The Hamming network is similar to the Hopfield network, introduced by R. Lippman (1987)
- Hamming network is two-network bipolar classifier.
- The first layer is single-layer perceptron. It calculates hamming distance between the vectors.
- The second layer is a fully connected recurrent network with m neurons (similar to the Hopfield network).

Structure of Hamming net

- The number of inputs is n, which is the dimensionality of the pattern space.
- The number of outputs is j, which is the number of patterns to store.
- **input layer** – a layer built with neurons, all of those neurons are connected to all of the network inputs;
- **output layer** – which is called MaxNet layer; the output of each neuron of this layer is connected to input of each neuron of this layer, besides every neuron of this layer is connected to exactly one neuron of the input layer (as in the picture above).



Hamming network working algorithm

$$X^1 = \left(x_1^1, \ldots, x_n^1\right) \;\; X^2 = \left(x_1^2, \ldots, x_n^2\right) \;\; X^m = \left(x_1^m, \ldots, x_n^m\right)$$

$$w_{ij} = x_i^j / 2, \; T_j = n/2$$

$$y_j = d_j$$

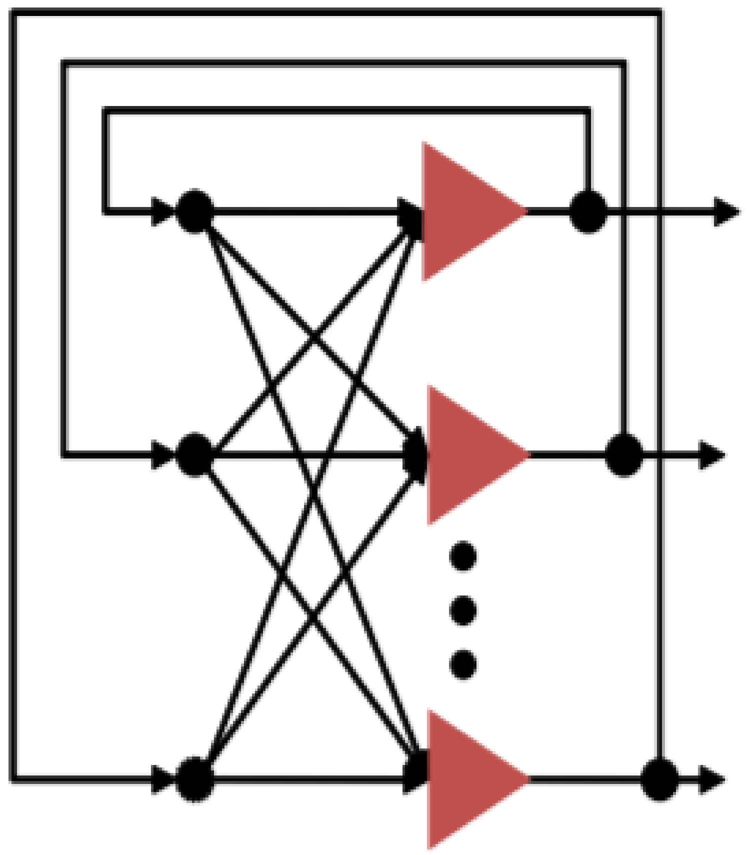
$$d_j - Hamming \;\; distance \; between \;\; input \; pattern \;\; and \;\; j \; stored \;\; pattern$$

- Define weights w_{ij}, T_j
- Get input pattern and initialize Hopfield weights
- Make iterations in Hopfield network until we get stable output.

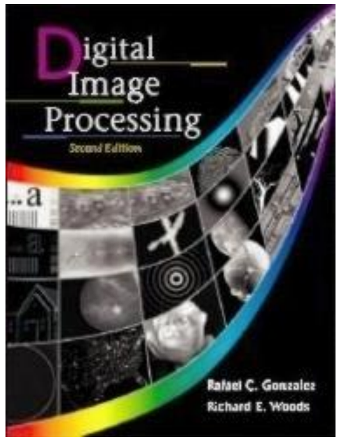
Hamming Neural Network

MAXNET: A Neural Network for Finding the Maximum Value

- MAXNET is similar to HNN.
- It is a single layer feed back neural network with *p* neurons.
- The diagonal elements of the weight matrix is always 1, and all other elements are –ε.
- The parameter ε is a constant in [0,1/p).
- The inputs (initial states) are real numbers in [0,1].



References



“Digital Image Processing”, Rafael C. Gonzalez & Richard E. Woods, Addison-Wesley, 2002

–“Fundamentals of Digital Image Processing” Anil K. Jain, 1989

–Image Processing and Pattern Recognition Slides of Dr. Sanjeeb Prasad Panday

–Neural networks Dr. Igor Anikin

Thank you !!!