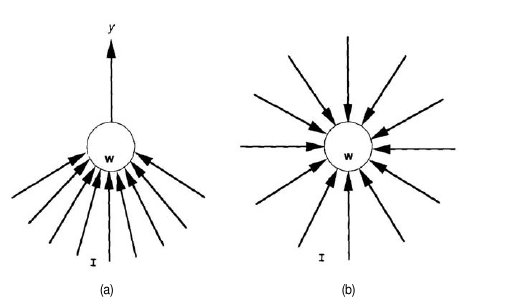
**The Instar**

The instar is a single processing element that shares its general structure and processing functions with many other processing elements. We distinguish it by the specific way in which it is trained to respond to input data.

Let's begin with a general processing element, such as the one in Figure 1.1(a). If the arrow representing the output is ignored, the processing element can be redrawn in the starlike configuration of Figure 1.1(b). The inward-pointing arrows identify the instar structure, but we restrict the use of the term instar to those units whose processing and learning are governed by the equations in this section.



**Figure 1.1**

*This figure shows:*

*(a) the general form of the processing element with input vector I, weight vector and output value and*

*(b) the form of the processing element in (a). Notice that the arrow representing the output is although it is still presumed to exist.*

The net-input value is calculated, as usual, by the dot product of the input and weight vectors, net = I •w. We shall assume that the input vector, I, and the weight vector, w, have been normalized to a length of 1. The output of the instar is governed by the equation

**Eq. (1.1)**  y = -ay + b net

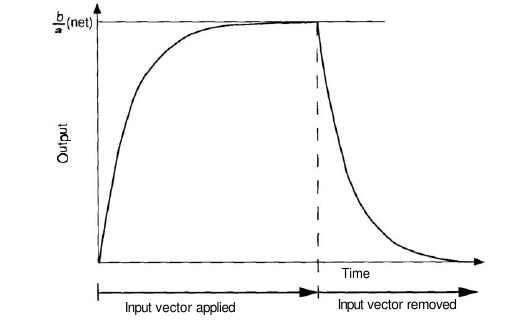
where a, b > 0. The dynamic behavior of y is illustrated in Figure 1.2.

We can solve Eq. (1.1) to get the output as a function of time. Assuming the initial output is zero, and that a nonzero input vector is present from time t = 0 until time t,

y(t) =

The equilibrium value of

yeq =



If the input vector is removed at time t`, then after equilibrium has been reached,

y(t) = yeq

for t > t`

**Figure 1.2**

This graph illustrates the output response of an instar. When an input vector is non-zero, the output rises to an equilibrium of (b/a)net. If the input vector is removed the output falls exponentially with a time constant of (1/a).

Notice that, for a given a and b, the output at equilibrium will be larger when the net-input value is larger.

lnstar is also known as Winner-take-all Learning Law. This is relevant for a collection of neurons, organized in a layer as shown in Figure 1.8. All the inputs are connected to each of the units

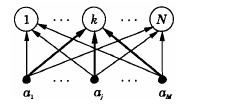


Figure 1.3

Arrangements of units for “instar learning”, where the adjusted weights are highlighted.

in the output layer in a feedfonvard manner. For a given input vector a, the output from each unit i is computed using the weighted sum WTIa. The unit k that gives maximum output is identified. That is

**WKTa = maxi (WiTa)**

Then the weight vector leading to the kth unit is adjusted as follows:



Therefore,



The final weight vector tends to represent a group of input vectors within a small neighbourhood. This is a case of unsupervised learning. In implementation, the values of the weight vectors are initialized to random values prior to learning, and the vector lengths are normalized during learning.

References

James A. Freeman. Neural networks for machine vision applications: The spacecraft orientation,Ford Aerospace pp. Fall 1988.

Stephen Grossberg. How does a brain build a cognitive code. In Stephen Grossberg, editor, Studies of Mind and Brain. D. Reidel Publishing, Boston, pp. 1-52, 1982.

Counterpropagation Networks James Freeman,

ARTIFICIAL NEURAL NETWORKS by B. Yegnanarayana, 1999.