Kohenen (SOM)

Simple example

Example 4.4 A Kohonen self-organizing map (SOM) to cluster four vectors

Let the vectors to be clustered be

$$(1, 1, 0, 0); (0, 0, 0, 1); (1, 0, 0, 0); (0, 0, 1, 1).$$

The maximum number of clusters to be formed is

$$m = 2.$$

Suppose the learning rate (geometric decrease) is

$$\alpha(0) = .6,$$

$$\alpha(t + 1) = .5\,\alpha(t).$$

With only two clusters available, the neighborhood of node J (Step *4)* is set so that only one cluster updates its weights at each step (i.e., R = *0)*.

Step 0. Initial weight matrix:

$$\begin{bmatrix} .2 & .8 \\ .6 & .4 \\ .5 & .7 \\ .9 & .3 \end{bmatrix}.$$

Initial radius:

$$R = 0.$$

Initial learning rate:

$$\alpha(0) = 0.6.$$

Step 1. Begin training.

Step 2. For the first vector, (1, *1*, 0, *0)*, do Steps 3–5.

Step 3.

$$D(1) = (.2 - 1)^2 + (.6 - 1)^2 + (.5 - 0)^2 + (.9 - 0)^2 = 1.86;$$

$$D(2) = (.8 - 1)^2 + (.4 - 1)^2 + (.7 - 0)^2 + (.3 - 0)^2 = 0.98.$$

Step 4. The input vector is closest to output node 2, so

$$J = 2.$$

Step 5. The weights on the winning unit are updated:

$$w_{i2}(\text{new}) = w_{i2}(\text{old}) + .6\,[x_i - w_{i2}(\text{old})]$$

$$= .4\,w_{i2}(\text{old}) + .6\,x_i.$$

Step 5. The weights on the winning unit are updated:

$$w_{i2}(\text{new}) = w_{i2}(\text{old}) + .6\,[x_i - w_{i2}(\text{old})]$$

$$= .4\, w_{i2}(\text{old}) + .6\, x_i.$$

This gives the weight matrix

$$\begin{bmatrix} .2 & .92 \\ .6 & .76 \\ .5 & .28 \\ .9 & .12 \end{bmatrix}.$$

Step 2. For the second vector, $(0, 0, 0, 1)$, do Steps 3–5.

Step 3.

$$D(1) = (.2 - 0)^2 + (.6 - 0)^2$$

$$+ (.5 - 0)^2 + (.9 - 1)^2 = 0.66;$$

$$D(2) = (.92 - 0)^2 + (.76 - 0)^2$$

$$+ (.28 - 0)^2 + (.12 - 1)^2 = 2.2768.$$

Step 4. The input vector is closest to output node *1*, so

XOR Classification

affine transformation from an input vector to an output scalar. Now, we describe an affine transformation from a vector $\boldsymbol{x}$ to a vector $\boldsymbol{h}$, so an entire vector of bias parameters is needed. The activation function g is typically chosen to be a function that is applied element-wise, with $h_i = g(\boldsymbol{x}^\top \boldsymbol{W}_{:,i} + c_i)$. In modern neural networks, the default recommendation is to use the *rectified linear unit* or ReLU (Jarrett *et al.*, 2009; Nair and Hinton, 2010; Glorot *et al.*, 2011a) defined by the activation function $g(z) = \max\{0, z\}$ depicted in Fig. 6.3.

We can now specify our complete network as

$$f(\boldsymbol{x}; \boldsymbol{W}, \boldsymbol{c}, \boldsymbol{w}, b) = \boldsymbol{w}^\top \max\{0, \boldsymbol{W}^\top \boldsymbol{x} + \boldsymbol{c}\} + b. \tag{6.3}$$

We can now specify a solution to the XOR problem. Let

$$\boldsymbol{W} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \tag{6.4}$$

$$\boldsymbol{c} = \begin{bmatrix} 0 \\ -1 \end{bmatrix}, \tag{6.5}$$

$$\boldsymbol{w} = \begin{bmatrix} 1 \\ -2 \end{bmatrix}, \tag{6.6}$$

and $b = 0$.

We can now walk through the way that the model processes a batch of inputs. Let $\boldsymbol{X}$ be the design matrix containing all four points in the binary input space, with one example per row:

$$\boldsymbol{X} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix}. \tag{6.7}$$

The first step in the neural network is to multiply the input matrix by the first layer's weight matrix:

$$\boldsymbol{XW} = \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 1 & 1 \\ 2 & 2 \end{bmatrix}. \tag{6.8}$$

Next, we add the bias vector $\boldsymbol{c}$, to obtain

$$\begin{bmatrix} 0 & -1 \\ 1 & 0 \\ 1 & 0 \\ 2 & 1 \end{bmatrix}. \tag{6.9}$$

In this space, all of the examples lie along a line with slope 1. As we move along this line, the output needs to begin at 0, then rise to 1, then drop back down to 0. A linear model cannot implement such a function. To finish computing the value of $\boldsymbol{h}$ for each example, we apply the rectified linear transformation:

$$\begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 1 & 0 \\ 2 & 1 \end{bmatrix}. \tag{6.10}$$

This transformation has changed the relationship between the examples. They no longer lie on a single line. As shown in Fig. 6.1, they now lie in a space where a linear model can solve the problem.

We finish by multiplying by the weight vector $\boldsymbol{w}$:

$$\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}. \tag{6.11}$$