Learning Vector Quantization Network

Vector Quantization

- If many patterns X_k cause cluster neuron j to fire with maximum activation a codebook vector $W_j = (w_{1j}, \ldots, w_{nj})^T$ behaves like a quantizing vector
- Quantizing vector : representative of all members of the cluster or class
- This process of representation is called vector quantization
- Principal Applications
  - signal compression
  - function approximation
  - image processing

Competitive Learning is Localized

- CL algorithms employ *localized learning*
  - update weights of only the active neuron(s)
- CL algorithms identify *codebook vectors* that represent invariant features of a cluster or class

Learning Vector Quantization Networks

- *Vector quantization*: an input space is divided into a number of distinct regions, and for each region a reconstruction vector is defined.
- A vector quantizer with minimum encoding distortion is called a *Voronoi* or nearest-neighbor quantizer.
- The collection of possible reproduction vectors is called the *code book* of the quantizer, and its members are called *code vectors*.

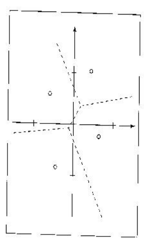

FIGURE 9.12 Voronoi diagram involving four cells. (Adapted from R.M Gray, 1984, with permission of IEEE.)

Learning Vector Quantizer

The SOM algorithm provides an approximate method for computing the Voronoi vectors in unsupervised manner.

Learning vector quantization (LVQ) is a supervised learning technique that uses class information to move the Voronoi vectors slightly, so as to improve the quality of the classifier decision regions.

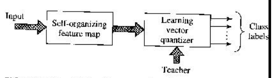

FIGURE 9.13 Block diagram of adaptive pattern classification, using a self-organizing feature map and learning vector quantizer.

Learning Vector Quantizer

- An input vector $\mathbf{x}$ is picked at random from the input space. If the class labels of the input vector $\mathbf{x}$ and a Voronoi vector $\mathbf{w}$ agree, the Voronoi vector $\mathbf{w}$ is moved in the direction of the input vector $\mathbf{x}$. If the class labels of the input vector $\mathbf{x}$ and the Voronoi vector $\mathbf{w}$ disagree, the Voronoi vector $\mathbf{w}$ is moved away from the input vector $\mathbf{x}$.
- Let $\{\mathbf{w}_i\}^{L}_{i=1}$ denote the set of Voronoi vectors, and the $\{\mathbf{x}_i\}^{N}_{i=1}$ denote the set of input vectors.

Learning Vector Quantizer

LVQ:

I. Suppose that the Voronoi vector $\mathbf{w}_c$ is the closest to the input vector $\mathbf{x}_i$. Let L_{wc} denote the class associated with the Voronoi vector $\mathbf{w}_c$ and L_{xi} denote the class label of the input vector $\mathbf{x}_i$. The Voronoi vector $\mathbf{w}_c$ is adjusted as follows:

If $L_{wc} = L_{xi}$,then $\mathbf{w}_c(n+1) = \mathbf{w}_c(n) + \alpha_n[\mathbf{x}_i - \mathbf{w}_c(n)]$ where $0< \alpha_n<1$.

If $L_{wc} \neq L_{xi}$,then $\mathbf{w}_c(n+1) = \mathbf{w}_c(n) - \alpha_n[\mathbf{x}_i - \mathbf{w}_c(n)]$ where $0< \alpha_n<1$.

II. The other Voronoi vectors are not modified.

LVQ Algorithm

$\mathbf{x}$	training vector $(x_1, \ldots, x_i, \ldots, x_n)$.
T	correct category or class for the training vector.
$\mathbf{w}_j$	weight vector for jth output unit $(w_{1j}, \ldots, w_{ij}, \ldots, w_{nj})$.
C_j	category or class represented by jth output unit.
$\lVert\mathbf{x} - \mathbf{w}_j\rVert$	Euclidean distance between input vector and (weight vector for) jth output unit.

Step 0. Initialize reference vectors (several strategies are discussed shortly); initialize learning rate, $\alpha(0)$.

Step 1. While stopping condition is false, do Steps 2–6.

Step 2. For each training input vector $\mathbf{x}$, do Steps 3–4.

Step 3. Find J so that $\lVert\mathbf{x} - \mathbf{w}_J\rVert$ is a minimum.

LVQ Algorithm

Step 4. Update $\mathbf{w}_J$ as follows:
if $T = C_J$, then

$$\mathbf{w}_J(\text{new}) = \mathbf{w}_J(\text{old}) + \alpha[\mathbf{x} - \mathbf{w}_J(\text{old})];$$

if $T \neq C_J$, then

$$\mathbf{w}_J(\text{new}) = \mathbf{w}_J(\text{old}) - \alpha[\mathbf{x} - \mathbf{w}_J(\text{old})].$$

Step 5. Reduce learning rate.

Step 6. Test stopping condition:
The condition may specify a fixed number of iterations (i.e., executions of Step 1) or the learning rate reaching a sufficiently small value.

LVQ Learning Example

VECTOR	CLASS
(1, 1, 0, 0)	1
(0, 0, 0, 1)	2
(0, 0, 1, 1)	2
(1, 0, 0, 0)	1
(0, 1, 1, 0)	2

The first two vectors will be used to initialize the two reference vectors. Thus, the first output unit represents class 1, the second class 2 (symbolically, $C_1 = 1$ and $C_2 = 2$). This leaves vectors (0, 0, 1, 1), (1, 0, 0, 0), and (0, 1, 1, 0) as the training vectors. Only one iteration (one epoch) is shown:

Step 0. Initialize weights:

$$\mathbf{w}_1 = (1, 1, 0, 0);$$

$$\mathbf{w}_2 = (0, 0, 0, 1).$$

Initialize the learning rate: $\alpha = .1$.

Step 1. Begin computations.

Step 2. For input vector $\mathbf{x} = (0, 0, 1, 1)$ with $T = 2$, do Steps 3–4.

Step 3. $J = 2$, since $\mathbf{x}$ is closer to $\mathbf{w}_2$ than to $\mathbf{w}_1$.

Step 4. Since $T = 2$ and $C_2 = 2$, update $\mathbf{w}_2$ as follows:

$$\mathbf{w}_2 = (0,0,0,1) + .1[(0,0,1,1) - (0,0,0,1)]$$
$$= (0,0,.1,1).$$

Step 2. For input vector $\mathbf{x} = (1, 0, 0, 0)$ with $T = 1$, do Steps 3–4.

Step 3. $J = 1$.

Step 4. Since $T = 1$ and $C_1 = 1$, update $\mathbf{w}_1$ as follows:

$$\mathbf{w}_1 = (1,1,0,0) + .1[(1,0,0,0) - (1,1,0,0)]$$
$$= (1,.9,0,0).$$

Step 2. For input vector $\mathbf{x} = (0, 1, 1, 0)$ with $T = 2$, do Steps 3–4.

Step 3. $J = 1$.

Step 4. Since $T = 2$, but $C_1 = 1$, update $\mathbf{w}_1$ as follows:

$$\mathbf{w}_1 = (1,.9,0,0) - .1[(0,1,1,0) - (1,.9,0,0)]$$
$$= (1.1,.89,-.1,0).$$

Consider an LVQ net with two input units and four target classes: C_1, C_2, C_3, and C_4. There are 16 classification units, with weight vectors indicated by the coordinates on the following chart, read in row-column order. For example, the unit with weight vector (0.2, 0.4) is assigned to represent Class 3, and the classification units for Class 1 have initial weight vectors of (0.2, 0.2), (0.2, 0.6), (0.6, 0.8), and (0.6, 0.4).

x_2							
1.0							
0.8		C_3	C_4	C_1	C_2		
0.6		C_1	C_2	C_3	C_4		
0.4		C_3	C_4	C_1	C_2		
0.2		C_1	C_2	C_3	C_4		
0.0							
	0.0	0.2	0.4	0.6	0.8	1.0	x_1

a. Present an input vector of (0.25, 0.25) representing Class 1. Using a learning rate of $\alpha = 0.5$, show which classification unit moves where (i.e., determine its new weight vector).

b. Present an input vector of (0.4, 0.35) representing Class 1. What happens?

c. Instead of presenting the second vector as in part b, present the vector (0.4, 0.45). What happens?