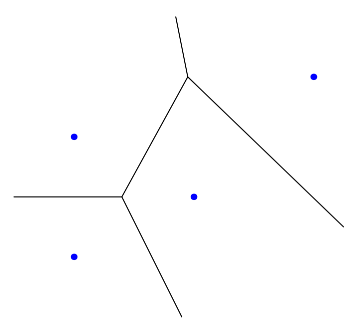
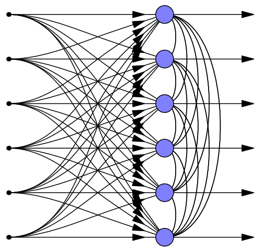
Competitive Learning

Vector Quantization

- Limited number of typical data vectors
- Compression: use the closest typical vector
- Voronoi-partitioning



Vector Quantization in Networks



- Winner-take-all
- Only one active output unit
  Compare: Grandmother-cells
- Possible biological counterpart: Lateral inhibition

Competitive Learning

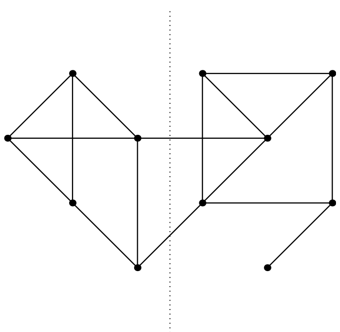
Basic Principle

Only update the winning unit (prototype vector) to become "even better".

Properties:

- Finds clusters in input data
- "Protects territories"
- Batch-version: $\vec{w} \leftarrow$ average of all $\vec{x}$ where $\vec{w}$ is a winner
- Reminds of Expectation Maximization (EM)

Example: graph partitioning



Split so that few edges cross the cut

Let every edge be a data point:

$$\vec{x} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \ldots$$

- Two output units $\Rightarrow$ Partition the edges in two groups
- May give an optimal solution

Dead-unit problem

Prototype vectors far from actual data will never become better

Methods to avoid dead units:

- Initialize using real samples
- "Umbrella unfolding" (start from one point)
- "Leaky learning" (some learning for all)
- Conscience (let losers win more easily)
- Noise, with a long tail
- Winners neighbors are also updated

LVQ — Learning Vector Quantization
Supervised Competitive Learning — Known classes

$$\Delta\vec{w} = +\eta(\vec{x} - \vec{w})$$

if the winner belongs to the right class

$$\Delta\vec{w} = -\eta(\vec{x} - \vec{w})$$

if the winner belongs to the wrong class