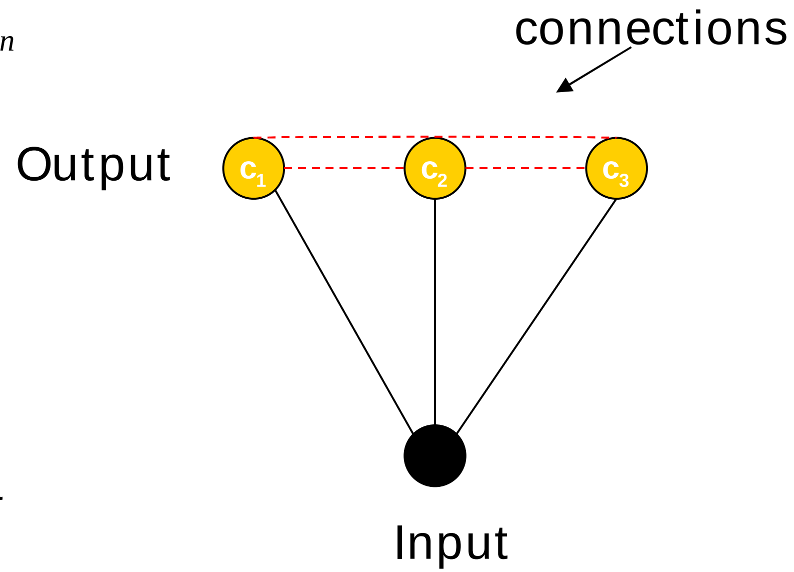
Competitive Learning

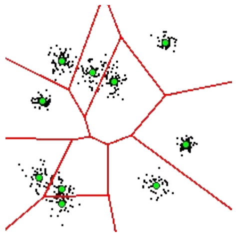
Self-organizing maps,
Neural Gas, etc.

Dizan Vasquez

Overview & properties

- N Units: $A = \{c_1, \ldots, c_N\}$
- Reference vectors: w_c in R^n
- Connections: C in $A \times A$ (maybe empty)
- Output units compete to turn on.
- Normally, learning minimizes distortion for a dataset *D*:

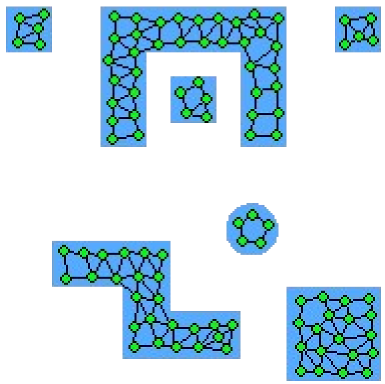
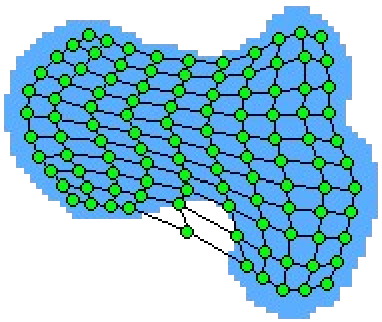
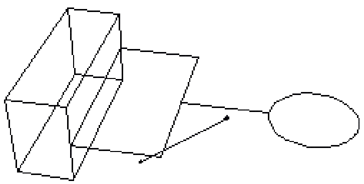
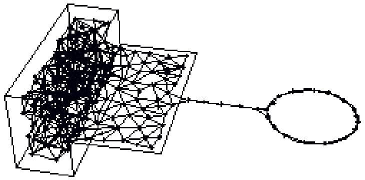
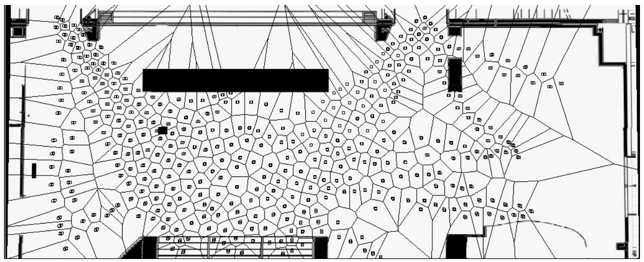




$$\mathrm{E}(\mathcal{D}, \mathcal{A}) = 1/|\mathcal{D}| \sum_{c \in \mathcal{A}} \sum_{\xi \in \mathcal{R}_c} \|\xi - \mathbf{w}_c\|^2$$

Some Uses

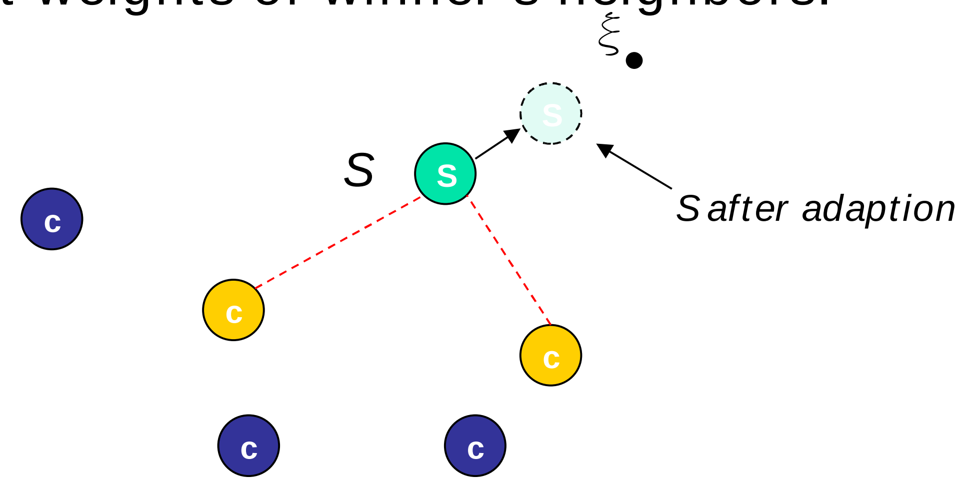
- Data compression
- Clustering
- Feature mapping
- Vector quantization
- Topology learning

General CL Algorithm

Input: Real vector ξ (signal)

2. Find winner S: $S = \arg\min_c \|\xi - w_c\|$
3. Adapt winner weight: $w_s = w_s + e(\xi - w_s)$
   e is called the learning rate.
4. For soft competitive learning:
   Adapt weights of winner's neighbors.



Algorithm Hierarchy

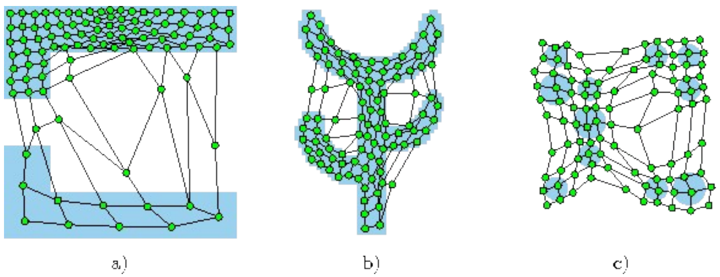
- Hard-competitive
  - Batch
    - LBG [Forgy65]
  - On-line
    - HCL
    - On-line k-Means [MacQueen67]
    - Exponential Decay [RitterEtAl91]
- Soft-competitive
  - Fixed Topology
    - Fixed N
      - Self-organizing map [Kohonen81]
    - Variable N
      - Growing Cell Structures [Fritzke94]
      - Growing Grid [Fritzke95]
  - Variable Topology
    - Fixed N
      - NG with Hebbian Learning [MartinetzShulten9
    - Variable N
      - Growing Neural Gas [Fritzke95]
      - Grow When Required [MarslandEtAl02]
  - No Topology
    - Neural Gas [MartinetzSchulten91]

Self-Organizing Map

Fixed topology, 2-dimensional grid.

Neighborhood is estimated by using a function:

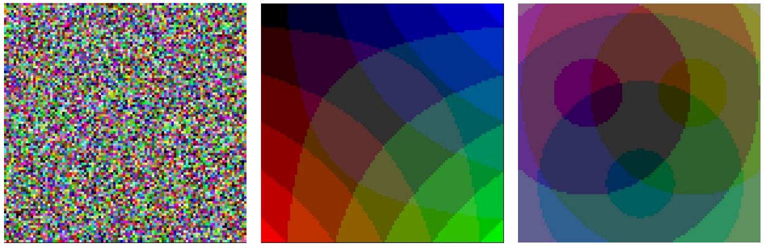
$$h_{rs} = \exp\left(\frac{-d_1(r,s)^2}{2\sigma^2}\right).$$



Standard deviation and learning rate decrease with time:

$$\sigma(t) = \sigma_i(\sigma_f/\sigma_i)^{t/t_{max}}$$

$$\epsilon(t) = \epsilon_i(\epsilon_f/\epsilon_i)^{t/t_{max}}.$$



Adaption:

$$w_r = w_r + e(t)\, h_{rs}\, (\xi - w_r)$$

Growing Neural Gas

Topology is found by the algorithm

Neighborhood is explicitly represented by links between units.

All parameters are constant

Adaptation:

Winner: $w_s = w_s + e_s(\xi - w_s)$

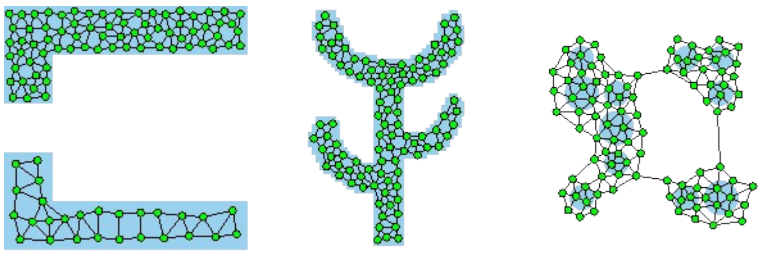
Neighbors $w_n = w_n + e_n(\xi - w_n)$

Error is cummulated for the winner:

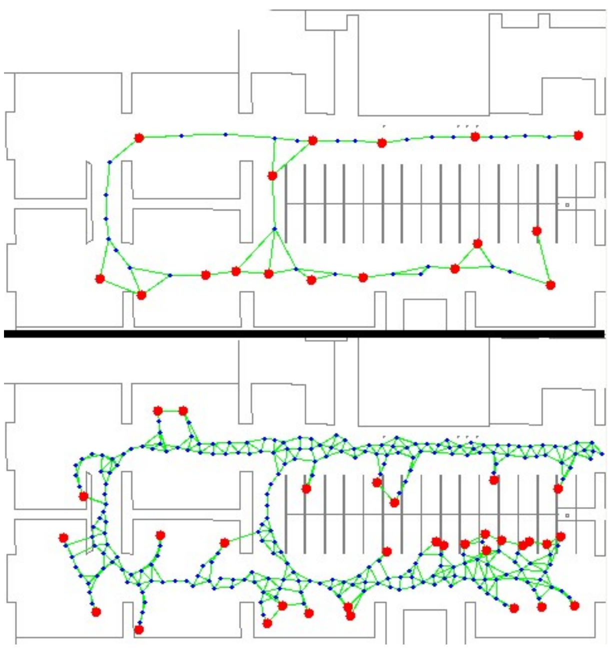
$$\Delta \mathbf{E}_{s_1} = \|\boldsymbol{\xi} - \mathbf{w}_{s_1}\|^2.$$

Starts with two units, a new unit is added every λ iterations near the unit having maximum cumulated error.

Old links are deleted, when a unit has no links, it is deleted.



a) b) c)



Conclusions

CL provides a mapping from continuous n-dimensional space to a discrete set of samples.

The resulting map may or may not have a topology which may be fixed or learned.

The number of units may also be fixed or learned from data.

Convergence is reached through local adaptations.

Hard CL adapts only the winner (winner takes all), while soft CL adapts also the neighbors.

Selection of the best algorithm depends heavily on the problem being studied.

Close links with other problems: clustering, vector quantization, radial base functions, etc.