Introduction to Neural Networks

Johns Hopkins University
Engineering for Professionals Program

605-447/625-438

Dr. Mark Fleischer

Module 12.3: Competitive Learning and Self-Organized Maps

In This Module We Will Cover:

- Competitive Learning
  - The Hamming Net
  - The MAXNET algorithm
  - Self-Organizing Maps
  - Data values ‘compete’ in some fashion

Self-Organizing Maps

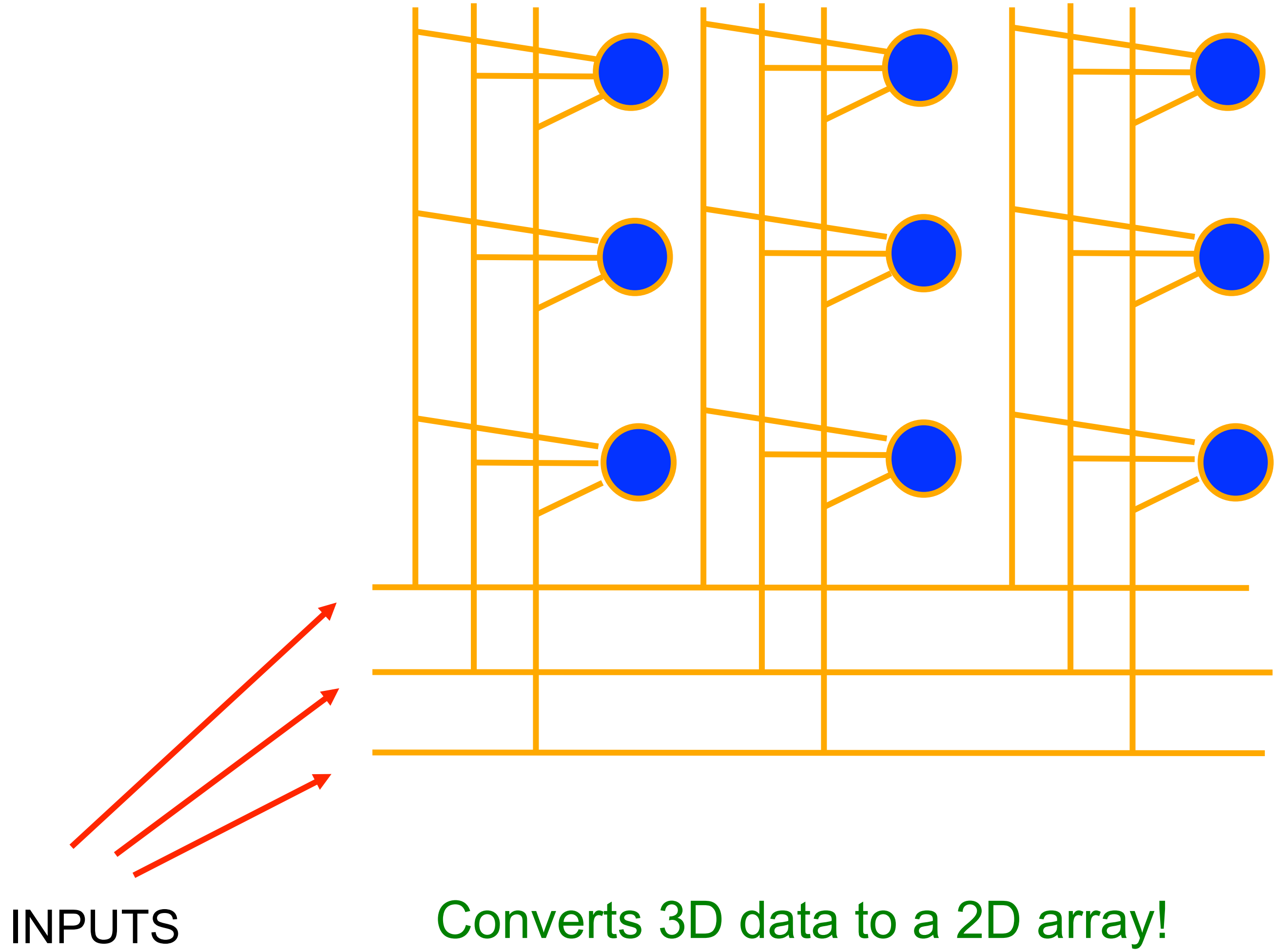
- Self-Organizing Maps (a.k.a. Kohonen Nets)
  - Unsupervised learning.
  - Data causes network to learn.
    - Data itself trains the net.
  - Uses Hamming Net and MAXNET.
  - Tries to create a *topology preserving map* where "near" inputs lead to "near" outputs.
  - Performs a type of feature extraction.
  - Akin to *operant conditioning*.
  - Data dimensionality reduction.
  - Displays a natural clustering behavior.

Self-Organizing Maps

What is the benefit of this?

- It extracts important features and segregates these features (see examples).
- It in effect allows for redundancy and fault tolerance.
- Reduces dimensions and displays similarities in input data. It can thus represent high-dimension data with smaller storage. The lack of dimension information is compensated in effect by a type of association topology, i.e., the feature extraction.

SOM Topology—an example



Converts 3D data to a 2D array!

Self-Organizing Maps

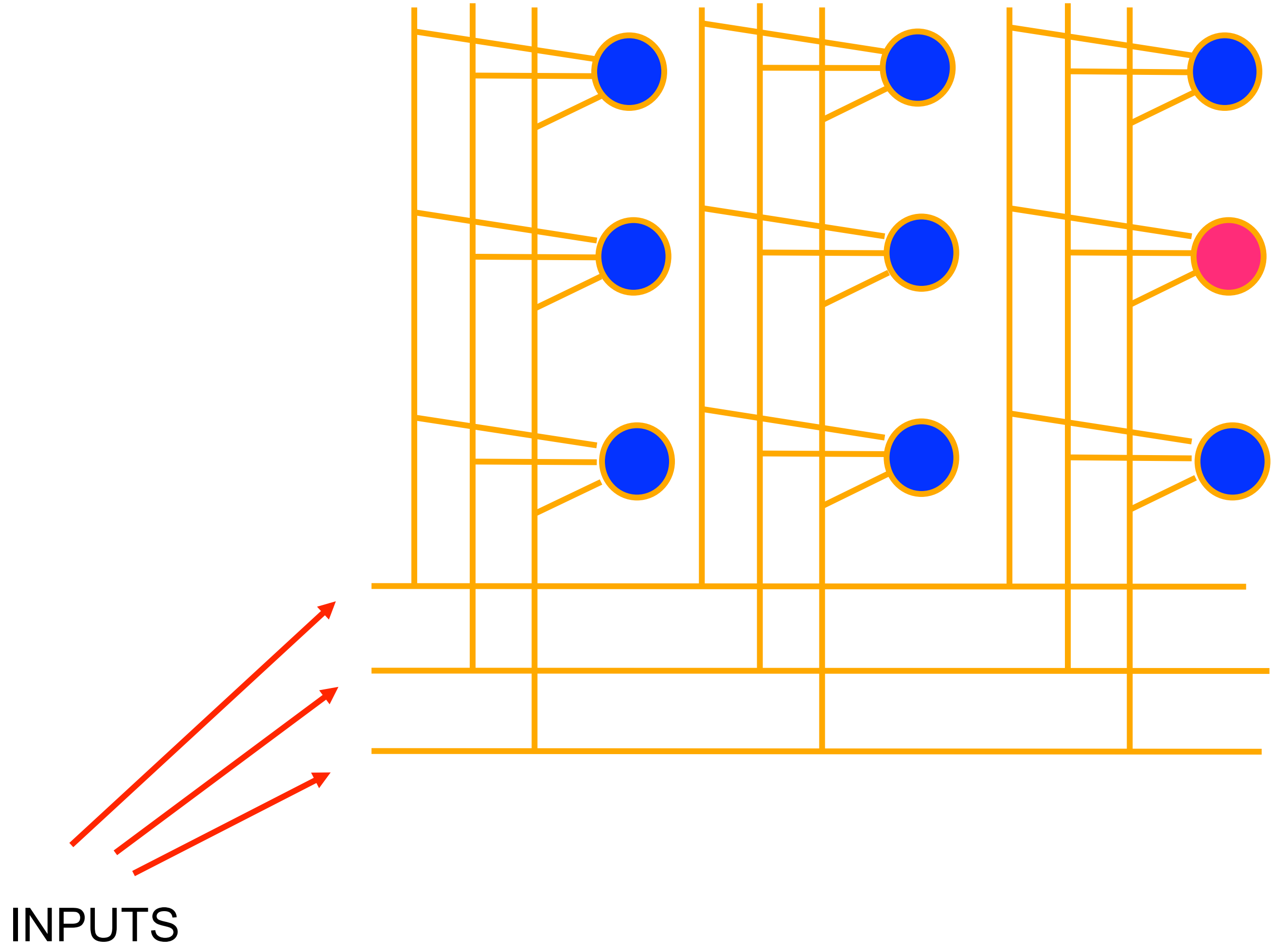
- Given this topology, each node has the 3 inputs **x** (shared with all other nodes) and a weight vector **w** (unique for each node) equal in size to the inputs. Basic idea is that for a given input (here a 3-tuple of possibly real values), with randomly assigned weights **w** for each node), determine which node's weights are 'closest' to the input vector.

Self-Organizing Maps

- Use competitive learning ala Hamming/ MAXNET to identify “winning” node.
- Use Hamming Distance, H, or other metrics: e.g.,

$$D(\mathbf{x},\mathbf{w}) = \left(\sum_{i=1}^{N} (x_i - w_i)^2 \right)^{1/2}$$

SOM Topology—an example

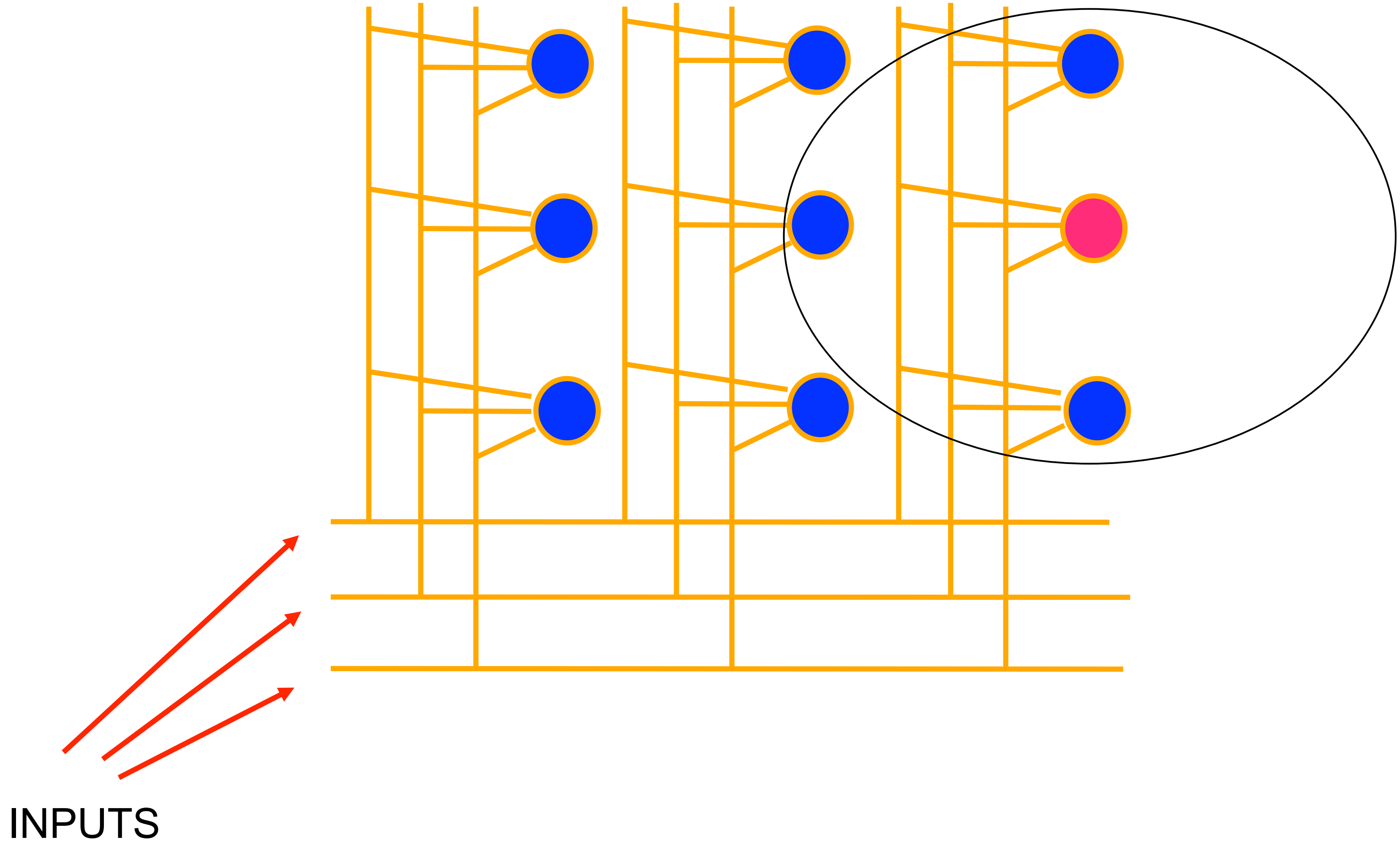


SOM Dynamics

- Select a neighborhood size σ and a neighborhood function *N*(*i,k*) associated with the winning node.
- This function determines the magnitude of changes to the weight vectors of neighboring nodes. For example, for a node *i*,

$$N(i,k) = e^{-|r_k - r_i|^2 / (2\sigma^2)}$$

SOM Topology—an example



SOM Dynamics

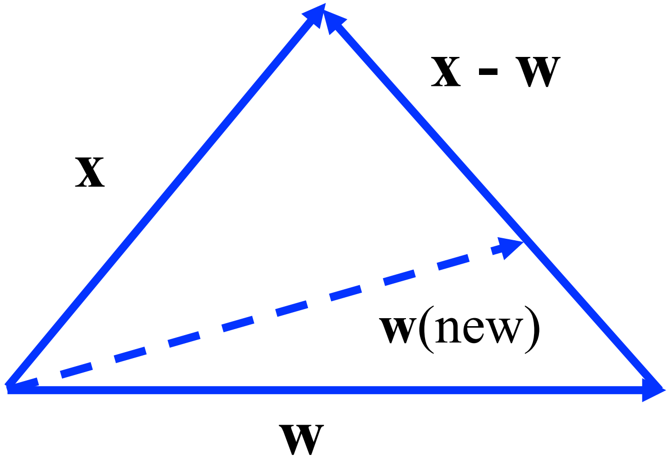
- Next we update all the weights of the neighboring (possibly all) nodes *k* in the array:

$$\mathbf{w}_k(\text{new}) = \mathbf{w}_k(\text{old}) + \mu N(i,k)(\mathbf{x} - \mathbf{w}_k)$$

What's going on here?
What does this update function do?

SOM Dynamics

- The weight vector is "moved" towards the input vector to a degree influenced by the topology (the nearness of the neighboring nodes):

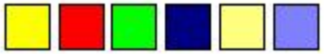


SOM Dynamics

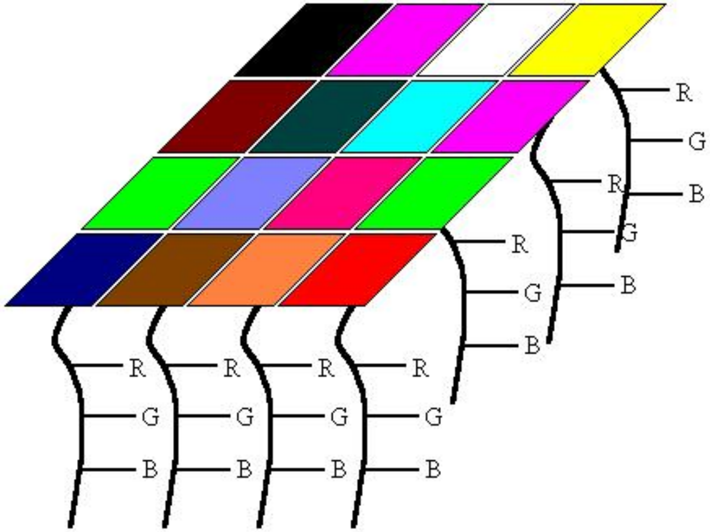
- Other similar update functions are possible, but the overall effect is to make the vector of weights closer to a given input pattern for the winning node and its neighbors.
- Overtime, the neighborhood weighting may lessen, the learning parameter also may decrease.
- Eventually, new input data gets mapped to a particular region of nodes with little if any effect on the nodes.
- Wanna see?

First some more background…

- **Example:** Use a 3 dimensional vector to represent colors in RGB:



- The network topology could be illustrated as:



First, Randomly Assign Weights

- In this example, this means randomly assign color values:

Other updating rules…

- Now update the nodes using an updating rule.
- Another, simpler rule is

$\mathbf{w}_{\mathbf{NEW}} = \mathbf{w}_{\mathbf{OLD}}\,(1 - \lambda) + \mathbf{x}(\lambda)$ where λ is initially 1

Eventually, this array becomes

Let’s see it in action…

http://davis.wpi.edu/~matt/courses/soms/applet.html

Other Examples….

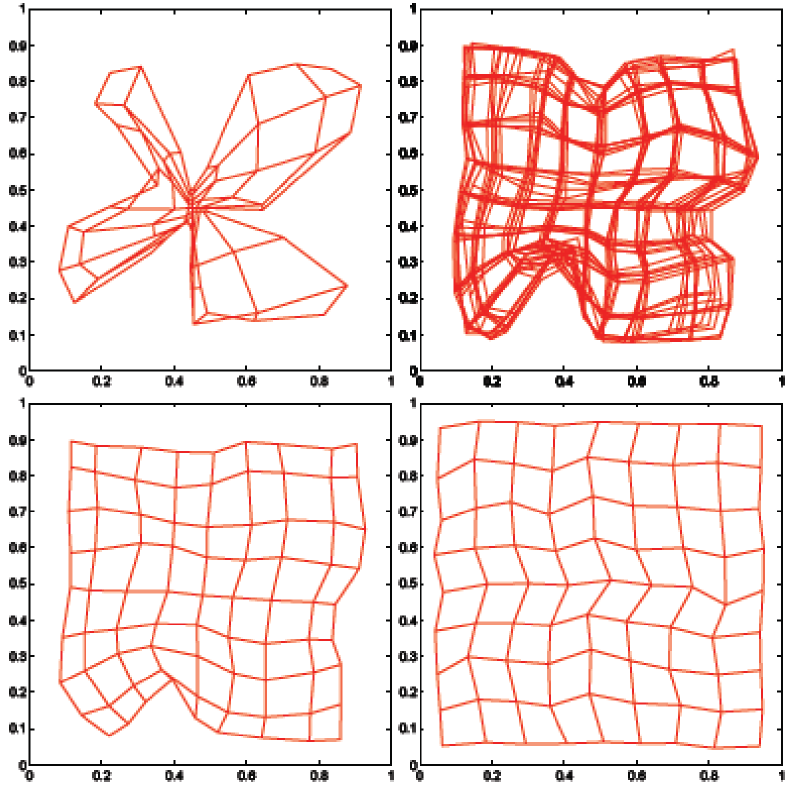


Fig. 15.7. Mapping a square with a two-dimensional lattice. The diagram on the upper right shows some overlapped iterations of the learning process. The diagram below it is the final state after 10000 iterations.

From Rojas.

Summary

- The Hamming Distance measure for discrete vectors was modified to define a metric that measures how well a discrete vector matches another discrete vector.
- Defined an iterative scheme such that a node that has the greatest activity value will be the only node in a set of nodes that has a positive value.
- Competitive Learning.
- Applied 3 dimensional color values to a 2 dimensional array.
- Applied 2 dimensional 'lattice values' to display a uniform distribution of input vectors.