Slide 0

Lecture 4.2

Kohonen's Self-organizing Feature Maps

Jonathan Shapiro

Department of Computer Science, University of Manchester

February 5, 2003

Slide 1

Motivation

Competitive learning has several problems.

Grandmother cell representation: Each input pattern causes a single node to fire. This is controversial.

1. Not robust to degradation.
2. Inconsistent with single-cell recording data, which is more consistent with cell-assembly representations.

Learning not stable: sensitive to weight initialization, dead units a problem, etc.

A solution — multiple grandmother cells arranged in geometry which respects neighborhoods in input pattern space

Slide 2

Self-organizing feature maps (SOFM)

The competitive learning algorithm is extended in three ways.

1. The neurons of the network are arranged in a 2-dimensional grid, or a line, or some other geometry.
2. During learning, not only are the weights of the winner updated, but the weights of the nearby nodes updated as well. There is a so-called neighborhood function which determines the size and shape of the neighborhood around the winning node. All of the nodes in this neighborhood have their weights updated along with the winning node.
3. As learning progresses, the size of the neighborhood function and the stepsize both gradually decrease.

Slide 3

Self-organizing map network architecture

- The network consists of a layer of neurons all receiving inputs, as with competitive learning.
- The location of the nodes within the network have some geometrical significance.

  Neighborhood function: $N(i, j)$ measures the distance between nodes i and j.

Slide 4

Kohonen's learning algorithm

Like competitive learning, but

1. Update weights of the winner and its neighbor. For all i, j,

$$\boxed{w_{ij} = w_{ij} + sN(i, i^*)\left[x_j - w_{i^* j}\right].}$$

2. Size of neighborhood shrinks during learning.
3. Stepsize s shrinks during learning.

Slide 5

What does the SOFM learn?

1. Topology preserving maps

- Topology means local or neighborhood properties.
- Attempts to map input patterns to nodes such that "nearness" is preserved; similar patterns mapped to nearby neurons. (Impossible in general, due to geometrical differences between input space and network geometry.
- Examples are found in biology — sensotopic maps.

  Retinotopic map: retina to visual cortex. Orientation selective cells arranged in a 2-d topology preserving map (Hubel and Wiesel 1962).

  Tonotopic map: a 1-d map arranged in frequency order.

Slide 6

2. More robust and hierarchical cluster

- Nodes more uniformly mapped across input patterns (i.e. dead units less likely)
- Clusters of clusters.

Slide 7

Examples

Edges revisited

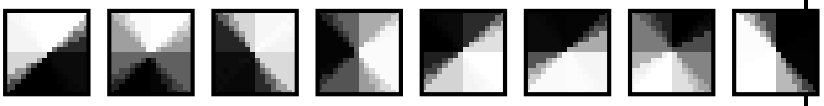

Figure 1: The weights of the 8 neurons in a 1-d Kohonen self-organizing map network. The network has discovered that there is a one-dimensional structure embedded in the 256 dimensional images – each image can be described by an angle.

Slide 8

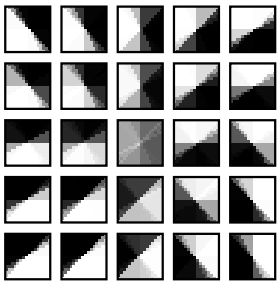

Figure 2: Same as above for a 5X5 2-d Kohonen self-organizing map network. Notice that neurons respond to similar images as those of the neurons next to them or above or below them.

Slide 9

Topological embedding movies

Slide 10

Phonemic typewriter

- Part of a Finnish speech recognition system
- Sampled at 13kHz
- Passed through low pass filter (5.3 kHz), 12 bit A/D, 256 point FFT, sliced into 9.83mS time windows
- Network input: 16-bit feature vector consisting of power spectrum of 9.83 mS time window

Note: phonemes are much longer, 40 – 400 mS.

Slide 11

points about phonemic typewriter

- Network “discovers” phonemes as important features (Kohonen claims).
- Figure shows nodes labelled subjectively by hand after training.
- Due to the topographic nature of the map, similar sounding phonemes cause nearby nodes to fire.
- If used for word recognition, errors will therefore produce similar sounding words to the original word.

Slide 12

Vector Quantization

Clustering algorithms such as Kohonen's SOM collapse a range of similar vectors onto a single prototype. This can be used for compression:

- Sender and receiver each have a copy of the trained neural network.
- Rather than transmitting the vector, transmit the *label* of the nearest prototype. Receiver then uses the prototype instead of the original vector.
- If the network is trained to represent the vectors well, this may give small loss in fidelity.