Self-Organizing Map
HW4 of Neural Network

YANG GUANG
1140339072
May 28, 2015

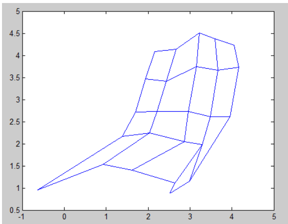
Introduction of SOM

Self-Organizing Map is trained using unsupervised learning to produce a low-dimensional (typically two-dimensional), discretized representation of the input space of the training samples, called a map. This makes SOMs useful for visualizing low-dimensional views of high-dimensional data, akin to multidimensional scaling. In this work we will focus on an Matlab implementation of SOM based on the given data source in the form of 5 X 5 neurons. The whole processing of SOM can be listed as follows:

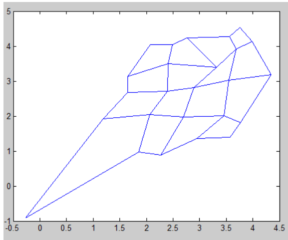
1. Initialize each node's weights
2. Choose a random vector from training data and present it to the SOM.
3. Every node is examined to find the Best Matching Unit (BMU). In our experiment, we just choose the node with minimal distance as the Best Matching Unit.
4. The radius of the neighborhood around the BMU is calculated. The size of the neighborhood decreases with each iteration.
5. Each node in the BMU's neighborhood has its weights adjusted to become more like the BMU. Nodes closest to the BMU are altered more than the nodes furthest away in the neighborhood. The updating rule can also be formulized as follows: $m_k = m_k + \alpha h(\| m_j - m_k \|)(x_j - m_k)$ where m_j is the BMU, m_k is the node to be updated, α is the learning rate, x_j is the data sample presented to SOM, h is the neighbor function which gives the similarity of m_j and m_k.
6. Repeat from step 2 for enough iterations for convergence.

Implementation and Result of SOM

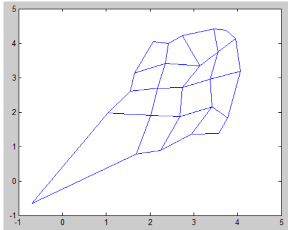
We using a 5 X 5 neurons with 2000 epoch as a maximum iteration. The folling figures are token from epoch 1, 10, 100, 1000 and 2000. We can clearly view a procedure of convergence.



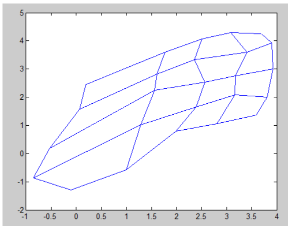
(a) 1 epoch



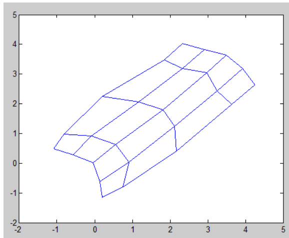
(b) 10 epoch



(c) 100 epoch



(d) 1000 epoch



(e) 2000 epoch

Figure 1: The coordinate of the centers