Self-Organizing Map on C++
Problem definition

Self-Organizing Maps, called SOMs from now on, are some fascinating tools for someone interested in dimensionality reduction. They were invented by a man named Teuvo Kohonen, a professor of the Academy of Finland, and they provide a way of representing multidimensional data in much lower dimensional spaces - usually one or two dimensions.

A SOM is a kind of artificial neural network (ANN) which 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, and is therefore a method to do dimensionality reduction. The difference between from the SOM and other ANN is that they apply competitive learning as opposed to error-correction learning (such as backpropagation with gradient descent), and in the sense that they use a neighborhood function to preserve the topological properties of the input space.

This process, of reducing the dimensionality of vectors, is essentially a data compression technique known as vector quantization. In addition, the Kohonen technique creates a network that stores information in such a way that any topological relationships within the training set are maintained.

I want to use these principles of SOMs while mapping the colors from their three-dimensional components - red, green and blue, into two dimensions. In **Fig. 1** it can be seen how a SOM is trained to recognize the eight different colors that are observed on the right-hand side. The network was given with the color representation as 3D vectors – each color component with its corresponding dimension – and the network learnt how to represent them in the 2D space as it can be noticed. An important remark would be that as a bonus to clustering the colors into different zones, the regions with similar properties are usually found adjacent to each other.

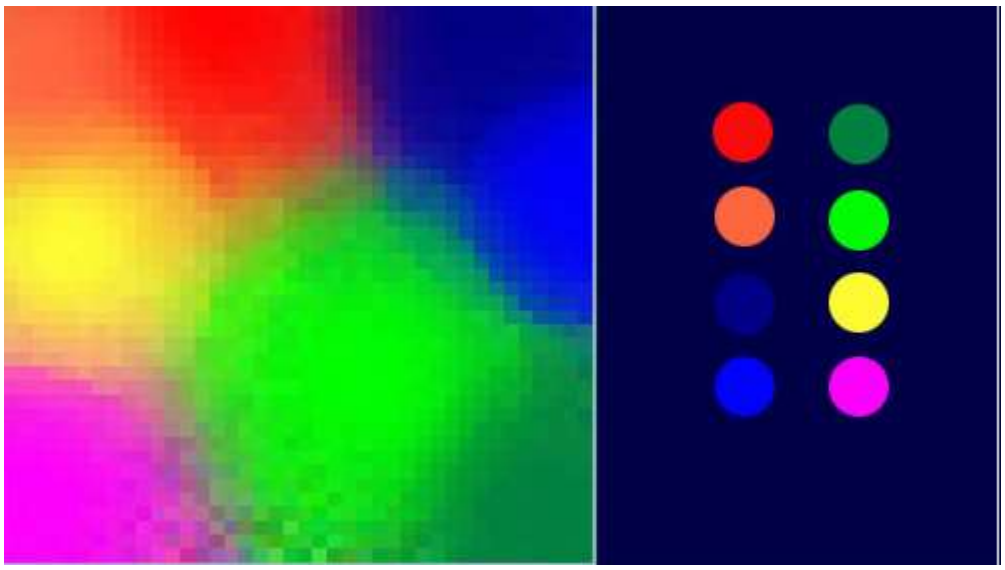

Fig. 1 Example of SOM

Network architecture

The network will be created from a 2D lattice of *nodes,* each of them being connected to the input layer. In **Fig. 2** a small network from the Kohonen perspective can be seen. The size is of 4 by 4 nodes. These nodes are connected to the input layer (the green one) which is a two-dimensional vector.

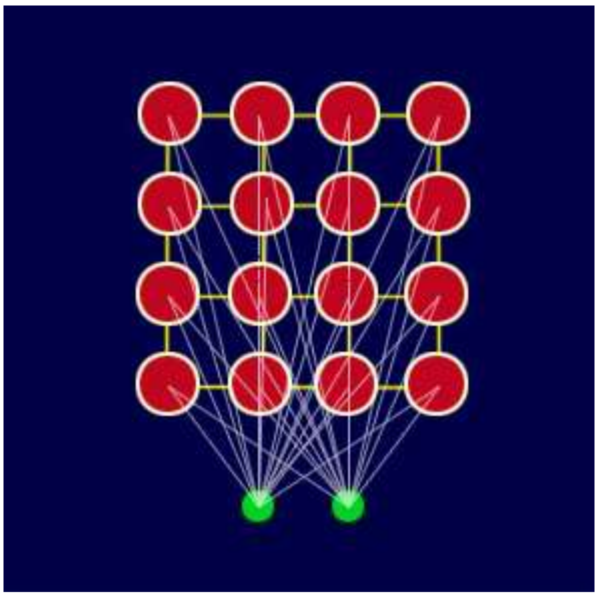

Fig. 2 A Kohonen Network

Each node comes with its specific topological position (x, y coordinate in the lattice) and contains a vector of weights of the same dimension as the input vectors. Mathematically speaking, if we consider the training data to be of vectors, V, of n dimensions, then we have:

$$V1, V2, V3 \ldots Vn$$

As a result, each node will contain a corresponding weight vector W, of n dimensions:

$$W1, W2, W3 \ldots Wn$$

Further discussion is needed so that everything related to this network is made clear. The purpose of the lines from **Fig. 2** that are connecting the nodes, is only to give us an idea about the adjacency and they have nothing to do with the usual connection used when dealing with a neural network. Within our considered lattice, there are no lateral connections between certain nodes.

Fig. 1 displays a SOM that has the default lattice size of 40 by 40. Three weights correspond to each of these nodes, one for each element of the input vector: red, green, blue. In the **Fig. 3** an easier representation is given, for further clearance.

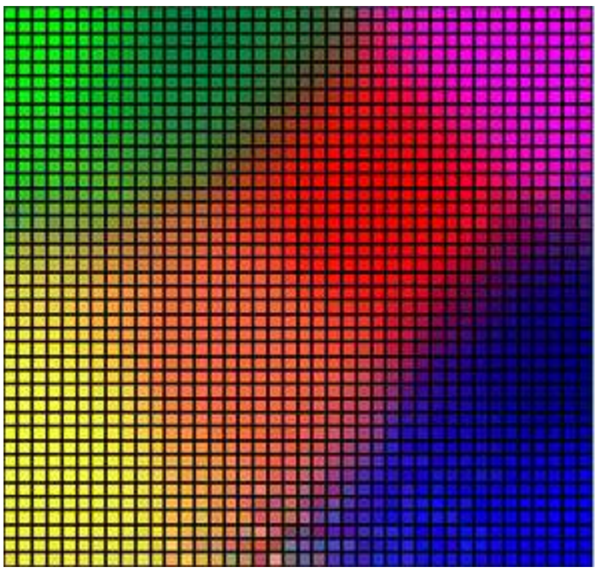

Fig. 3 The lattice of SOM