Kohonen's Self Organizing Map: Reduction of Dimension

Akira Imada

Most recently renewed on

December 7, 2007

1 Dimension reduction from 2-D to 1-D

We can use the Kohonen's SOM for dimension reduction. It's important because, we, human being couldnt never imagine the space whose dimension is more than three. In this section we study, as a toy example, how to map 2-D world into 1-D World. Also we are going to study how we can enjoy this trivial-looking mapping.

1.1 Let's construct a framework.

First of all, try the following as the very 1st excersise.

Excersise 1 (Winner takes all) *As an example of mapping from 2D to 1D, construct a SOM with two inputs and 100 outputs as follows:*

(1) All of the two input neurons connect to all of 100 output neurons. Thus, we have 200 syanapses. (See Figure 1.)

(2) Give each of all these 200 syanapses a random weight value between 0.00 to 1.00.

(3) Design a man-machine interface to show on the display screen how Winner-Take-All works.

- *(i) Display 100 output neurons in a 10 × 10 rectanglar array on the display screen.*
- *(ii) Disign so that the algorithm can accept two inputs x_1 and x_2 manually from your key-board. Assume x_1 and x_2 will be taken from 0.00 to 1.00.*
- *(iii) Make the winner highlited where the winner is the neuron which has the synaptic weights (w_1, w_2) closest distance to the input (x_1, x_2). Mesure the distance by Eucledean distance. That is,*

$$\sqrt{(x_1 - w_1)^2 + (x_2 - w_2)^2}$$

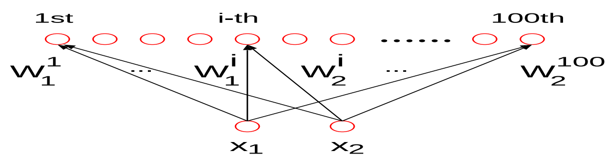


Figure 1: A framework of Kohonen's SOM for a mapping from 2-D to 1-D space.

1.2 Let's observe a dimension reduction?

Excersise 2 (Winner takes all) *Use the frame-work created in the previous week, that is, SOM with two inputs and 100 outputs, which realizes a mapping from 2D to 1D. Then instead of input from key-board, give 50 inputs automatically in your program. Input points should be chosen, for example, from incide of the unit-circle, (ii) rectangle or (iii) triangle. (See Figure 2.)*

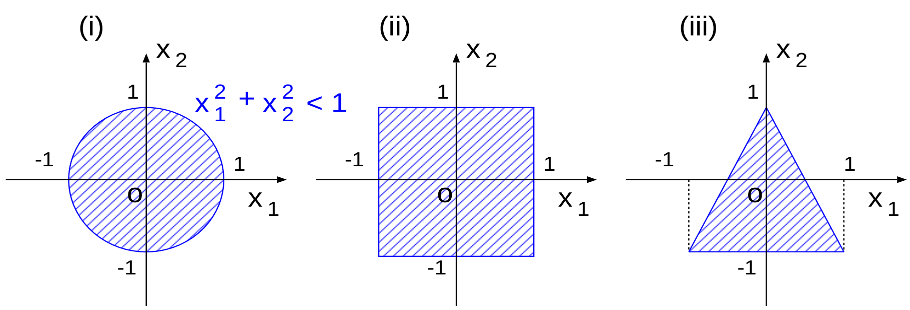


Figure 2: A dimension reduction from 2D to 1D still with randome weights.

1.3 How pairs of weights are distributed?

Excersise 3 (Distribution of a pair of weights) *Under the framework created in the Excersise 1 try the following experiment.*

(1) Plot all the weight pairs those determined at random in the Excersise 1 in 2D space. Show the result in the display screen.

(2) Design your code so that waight pair of the winner for a input from key-board is highlited. Demonstrate it on the screen.

1.4 Let's renew the weights of the winner neuron.

Excersise 4 (Renewal of weights) *Let's renew the weights of the winner neuron in the following way.*

(1) Give an input (x1, x2) randomly from keyboard

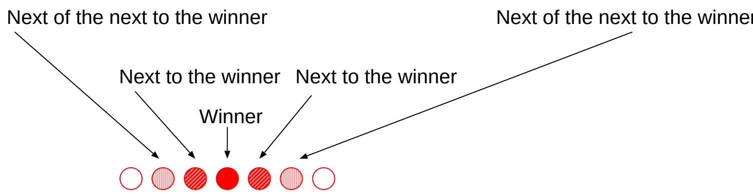


Figure 3: The winner neuron and its four neighbours.

(2) *Renew weights only of the winner neuron, the two neurons next to the winners, and and the next of the next (two neurons apart from the winner neuron. Hence, a total of* $2 \times 5 = 10$ *weights. with the equation*

$$\mathbf{w}_{new} = \mathbf{w}_{old} + g * K * (\mathbf{x} - \mathbf{w}_{old}) \tag{1}$$

where g *is a strength of infuluence of renew. Strength of the modification of the weights is the strongest at the winner, then gradually become weak and if the neuron is more than two neurons far from the winner, weights will be no more modified.* [1] *That is, (1) the winner should be strongly influenced while (2) the next two neighbors should be not so strong, and (3) the next of the next neighbors should have a weaker influence than the next ones. So, try* $g = 1$ *for the winner,* $g = 0.5$ *for the next two, and* $g = 0.25$ *for the next next.*

(3) *Then show on your display 5 weight pair with winner being center the two next neighbors both sides and two next next neighbors also both sides, like the below:*

$$(w^{old}_{1\to i-2}, w^{old}_{2\to i-2})(w^{old}_{1\to i-1}, w^{old}_{2\to i-1})(w^{old}_{1\to i}, w^{old}_{2\to i})(w^{old}_{1\to i+1}, w^{old}_{2\to i+1})(w^{old}_{1\to i+2}, w^{old}_{2\to i+2})$$

$$(w^{new}_{1\to i-2}, w^{new}_{2\to i-2})(w^{new}_{1\to i-1}, w^{new}_{2\to i-1})(w^{new}_{1\to i}, w^{new}_{2\to i})(w^{new}_{1\to i+1}, w^{new}_{2\to i+1})(w^{new}_{1\to i+2}, w^{new}_{2\to i+2})$$

Figure 4: The winner neuron and its four neighbours.

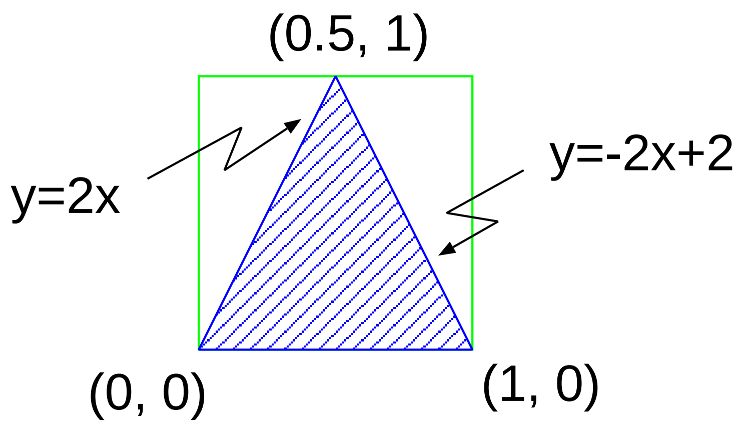
1.5 Self Organization of Weights

We now experiment a self-organization of weights values. Try the below.

Excersise 5 (A Self Organization of Weights) *Instead of inputs from the keyboard we give our Kohonen network a set of many inputs, for example, from inside a triangle.*

[1] The first neuron doesn't have next neuron and the next of the next neuron of its left hand side. Also the second neuron only has the next neuron to its left hand side. The same holds the second last neuron and the last neurons.

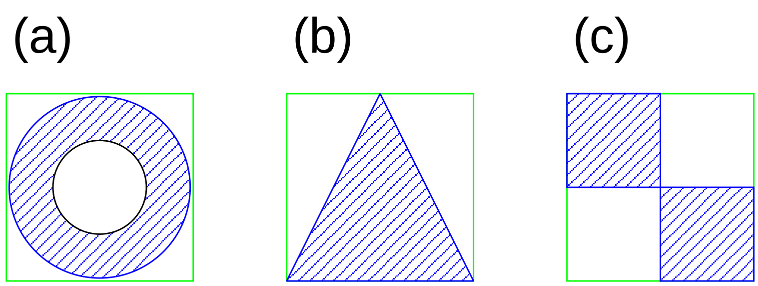
(1) Pick up one point from inside of the triangle whose three tops are (0,0), (1,0) and (0.5, 1)[2]



(2) Renew weights.

(3) Repeat (1) and (2) 30,000 times with $K = 0.4$ *for the 1st 10,000 iteration,* $K = 0.2$ *from the next 10,000 iteration, and* $K = 0.1$ *for the last 10,000 iteration*

Or, more in general, try inputs from the followings



[2] If $x_1 < 0.5$ and $x_2 < 2x_1$ then point (x_1, x_2) is inside the triangle, else if $x_1 > 0.5$ and $x_2 < -2x_1 + 2$ then point (x_1, x_2) is inside the triangle.

1.6 A Result and its Interpretation

The bellow is an example of the result of plotting

$$(w_1^1, w_2^1), (w_1^2, w_2^2), \cdots, (w_1^{100}, w_2^{100})$$

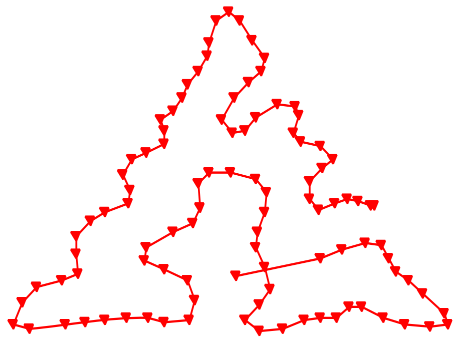


Figure 5: 100 pairs of weight linked with a line from point No.1 to point No.100. Note that this look like a *Peano curve*.

However, note that this is not a result of *dimension reduction* but a result of *self-organization of weight values.* It would be easy to understand if you think that dimensionality of weight space of n-D to n-D mapping is also n-D.

1.7 Let's see an actual mapping from 2D to 1D

Then what on earth is the result of *demension reduction?* See the Figure below.

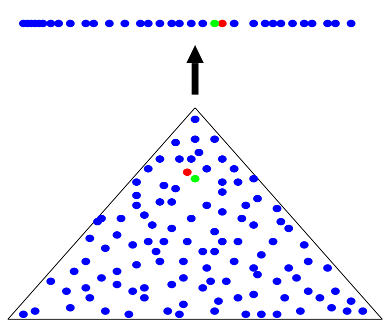


Figure 6: Maping from points in a triangle in 2-D to 1-D. A color in a circle corresponds each other between 1-D and 2-D.

1.8 A More Practical Application from 2D to 1D – A Thought Experiment

If you map a coloured map made up of pikcels defined by RGB color element. That is, each point of source space needs 5 parameters – 2 coordinates for location, abd 3 RGB values. Then mapping them implies to 1D but each point mapped also have an information of RGB colors.

This might be better to say mapping from 5D to 4D, though visually it is mapping to 1D. Of course similar colors should be mapped closer than others.

Try a *Thought Experiment* how the mapping of points including RGB information in Figure 7 from 2D space to a 1D space.

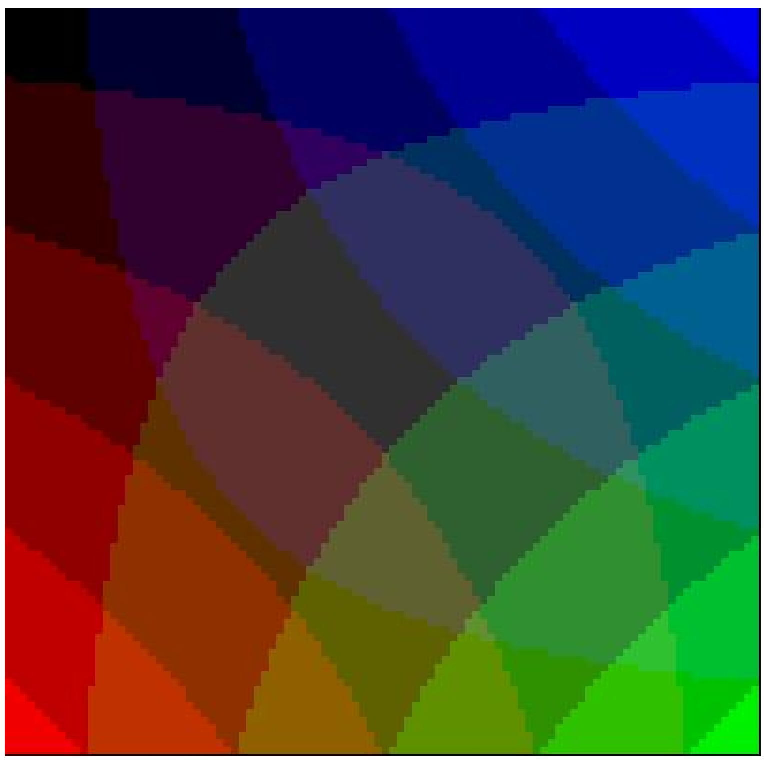


Figure 7: A source of RGB color map visually in 2-D but actually implying 6-D.

2 Demension Reduction from 2-D to 2-D

This section title sounds strange in the sense that from 2-D to 2-D would not be a reduction. Actually, it will be a tribial example but, nevertheless, it will be interesting to observe what will be happen to plotted pair of weights which map from the input 2-D space to lattice grid in the output 2-D space.

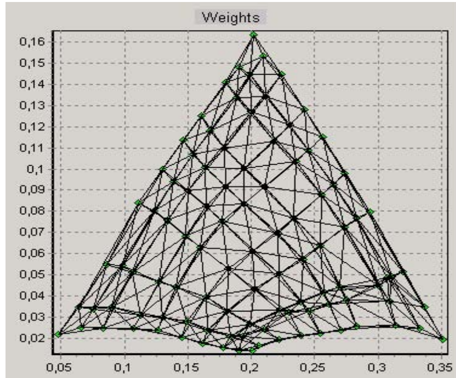


Figure 8: Self-organized weights of 100 synaptics from 2 input to 10x10 outputs. 10,000 random points are repeatedly given. — From a student's result during the course. By Yakushevich Maxim (Fall Semester in 2007).

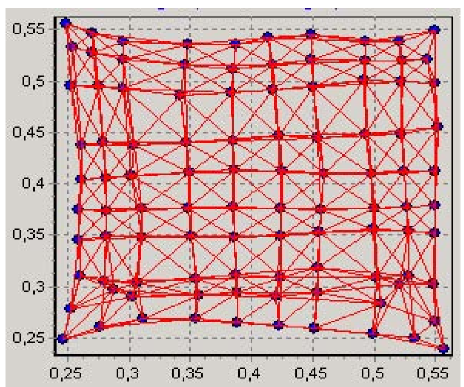


Figure 9: Self-organized weights of 100 synaptics from 2 input to 10x10 outputs. 10,000 random points are repeatedly given. — From a student's result during the course. By Novikov Andrei. Fall Semester in 2007

3 From more than 3D into 2D

Iris flower dataset[3] is made up of 150 samples consists of three species of iris flower, that is, *setosa, versicolor* and *virginica*. Each of these three families includes 50 samples. Each sample is a four-dimensional vector representing four attributes of the iris flower, that is, *sepal-length, sepal-width, petal-length*, and *petal-width*.

Let's visualize these 150 points in 4-D space, by reducing the dimension to 2D. The data is as follows in the next two pages. Plot 150 points in 2-D space using 3 different colors from one specie to the other.

[3]This can be obtained from University of California Urvine Machine Learning Repository. ftp://ics.uci.edu: pub/machine-learning-databases.

Setosa				Versicolor				Virginica			
x_1	x_2	x_3	x_4	x_1	x_2	x_3	x_4	x_1	x_2	x_3	x_4
0.65	0.80	0.20	0.08	0.89	0.73	0.68	0.56	0.80	0.75	0.87	1.00
0.62	0.68	0.20	0.08	0.81	0.73	0.65	0.60	0.73	0.61	0.74	0.76
0.59	0.73	0.19	0.08	0.87	0.70	0.71	0.60	0.90	0.68	0.86	0.84
0.58	0.70	0.22	0.08	0.70	0.52	0.58	0.52	0.80	0.66	0.81	0.72
0.63	0.82	0.20	0.08	0.82	0.64	0.67	0.60	0.82	0.68	0.84	0.88
0.68	0.89	0.25	0.16	0.72	0.64	0.65	0.52	0.96	0.68	0.96	0.84
0.58	0.77	0.20	0.12	0.80	0.75	0.68	0.64	0.62	0.57	0.65	0.68
0.63	0.77	0.22	0.08	0.62	0.55	0.48	0.40	0.92	0.66	0.91	0.72
0.56	0.66	0.20	0.08	0.84	0.66	0.67	0.52	0.85	0.57	0.84	0.72
0.62	0.70	0.22	0.04	0.66	0.61	0.57	0.56	0.91	0.82	0.88	1.00
0.68	0.84	0.22	0.08	0.63	0.45	0.51	0.40	0.82	0.73	0.74	0.80
0.61	0.77	0.23	0.08	0.75	0.68	0.61	0.60	0.81	0.61	0.77	0.76
0.61	0.68	0.20	0.04	0.76	0.50	0.58	0.40	0.86	0.68	0.80	0.84
0.54	0.68	0.16	0.04	0.77	0.66	0.68	0.56	0.72	0.57	0.72	0.80
0.73	0.91	0.17	0.08	0.71	0.66	0.52	0.52	0.73	0.64	0.74	0.96
0.72	1.00	0.22	0.16	0.85	0.70	0.64	0.56	0.81	0.73	0.77	0.92
0.68	0.89	0.19	0.16	0.71	0.68	0.65	0.60	0.82	0.68	0.80	0.72
0.65	0.80	0.20	0.12	0.73	0.61	0.59	0.40	0.97	0.86	0.97	0.88
0.72	0.86	0.25	0.12	0.78	0.50	0.65	0.60	0.97	0.59	1.00	0.92
0.65	0.86	0.22	0.12	0.71	0.57	0.57	0.44	0.76	0.50	0.72	0.60
0.68	0.77	0.25	0.08	0.75	0.73	0.70	0.72	0.87	0.73	0.83	0.92
0.65	0.84	0.22	0.16	0.77	0.64	0.58	0.52	0.71	0.64	0.71	0.80
0.58	0.82	0.14	0.08	0.80	0.57	0.71	0.60	0.97	0.64	0.97	0.80
0.65	0.75	0.25	0.20	0.77	0.64	0.68	0.48	0.80	0.61	0.71	0.72
0.61	0.77	0.28	0.08	0.81	0.66	0.62	0.52	0.85	0.75	0.83	0.84
0.63	0.68	0.23	0.08	0.84	0.68	0.64	0.56	0.91	0.73	0.87	0.72
0.63	0.77	0.23	0.16	0.86	0.64	0.70	0.56	0.78	0.64	0.70	0.72
0.66	0.80	0.22	0.08	0.85	0.68	0.72	0.68	0.77	0.68	0.71	0.72
0.66	0.77	0.20	0.08	0.76	0.66	0.65	0.60	0.81	0.64	0.81	0.84
0.59	0.73	0.23	0.08	0.72	0.59	0.51	0.40	0.91	0.68	0.84	0.64
0.61	0.70	0.23	0.08	0.70	0.55	0.55	0.44	0.94	0.64	0.88	0.76
0.68	0.77	0.22	0.16	0.70	0.55	0.54	0.40	1.00	0.86	0.93	0.80
0.66	0.93	0.22	0.04	0.73	0.61	0.57	0.48	0.81	0.64	0.81	0.88
0.70	0.95	0.20	0.08	0.76	0.61	0.74	0.64	0.80	0.64	0.74	0.60
0.62	0.70	0.22	0.04	0.68	0.68	0.65	0.60	0.77	0.59	0.81	0.56
0.63	0.73	0.17	0.08	0.76	0.77	0.65	0.64	0.97	0.68	0.88	0.92
0.70	0.80	0.19	0.08	0.85	0.70	0.68	0.60	0.80	0.77	0.81	0.96

(to be cont'd to the next page)

(cont'd)

Setosa				Versicolor				Virginica			
x_1	x_2	x_3	x_4	x_1	x_2	x_3	x_4	x_1	x_2	x_3	x_4
0.62	0.70	0.22	0.04	0.80	0.52	0.64	0.52	0.81	0.70	0.80	0.72
0.56	0.68	0.19	0.08	0.71	0.68	0.59	0.52	0.76	0.68	0.70	0.72
0.65	0.77	0.22	0.08	0.70	0.57	0.58	0.52	0.87	0.70	0.78	0.84
0.63	0.80	0.19	0.12	0.70	0.59	0.64	0.48	0.85	0.70	0.81	0.96
0.57	0.52	0.19	0.12	0.77	0.68	0.67	0.56	0.87	0.70	0.74	0.92
0.56	0.73	0.19	0.08	0.73	0.59	0.58	0.48	0.73	0.61	0.74	0.76
0.63	0.80	0.23	0.24	0.63	0.52	0.48	0.40	0.86	0.73	0.86	0.92
0.65	0.86	0.28	0.16	0.71	0.61	0.61	0.52	0.85	0.75	0.83	1.00
0.61	0.68	0.20	0.12	0.72	0.68	0.61	0.48	0.85	0.68	0.75	0.92
0.65	0.86	0.23	0.08	0.72	0.66	0.61	0.52	0.80	0.57	0.72	0.76
0.58	0.73	0.20	0.08	0.78	0.66	0.62	0.52	0.82	0.68	0.75	0.80
0.67	0.84	0.22	0.08	0.65	0.57	0.43	0.44	0.78	0.77	0.78	0.92
0.63	0.75	0.20	0.08	0.72	0.64	0.59	0.52	0.75	0.68	0.74	0.72