

Understanding Dynamics of Kohonen's Self-Organizing Feature Maps

4358

April 22, 2003

Abstract

The aim of this work is to study the basics of an unsupervised learning method, Kohonen's algorithm, which works on top of a neural network whose units are geometrically related and activated in a winner-take-all basis. The network is capable of discovering and mapping the topological features in input patterns only using internal dynamics. Kohonen's Self-Organizing Map (SOM) is trained by a number of data sets which differ in their distribution on input space. One of the main objectives of this study is to optimize the parameters of the system while other objective is to analyze spatial ordering of output neurons with different data sets.

1 Introduction

Artificial Neural Network (ANN) modelling techniques have been used intensively in order to process information for decades. Researchers in this field approach to the problem of *creation of intelligent processing units* from a different viewpoint, they try to construct systems which mimic the brain functionality of higher animals. Modern neural science has a very strong argument which states that "*behavior is a reflection of brain function*" [1]. Thus, it is thought to be possible to obtain intelligent agents by understanding the internal representation and flow of knowledge within human nervous system and applying this in artificially created computational models. ANNs are composed of neuron-like units which activates each other through connections with adjustable

strengths/weights.

There are a number of machine learning techniques which can be classified into three groups. Supervised and reinforcement learning can be used only if an external trainer exists and gives directions during the training phase. The third group consists of models and algorithms, that are used in the lack of an external teacher. Unsupervised learning methods are applied to tasks, where a solution to the problem could emerge without any environmental feed-back.

In this paper, I will study on an unsupervised learning algorithm, known as Kohonen's algorithm [2], which works on top of a neural network whose units are geometrically related. The network is capable of discovering and mapping the topological features in input patterns only using internal dynamics. This networks are known as *Self-Organizing Feature Maps* which compose of competitive output-layer neurons, organized in an m-dimensional grid. Although details of the network and learning algorithm will be given in Section 2, it is worth to mention that feature in the input pattern is discovered and represented by the spatial locations of output neurons.

In section 3 the results of a number of experiments and discussions on the effectiveness of the algorithm will be presented. The first set of experiments are mainly conducted to understand the effect of parameters to the system and to optimize them. In the second set of experiments, parameters are adjusted dynamically during the training phase. Then, distribution of data in the training set is changed and the spatial ordering of output neurons are examined accordingly.

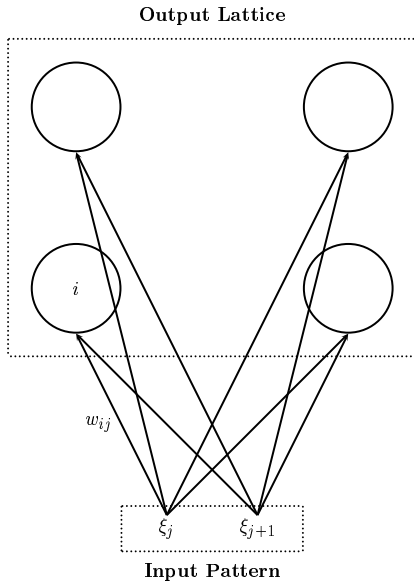


Figure 1: A sample full-connected feature mapping architecture. In this network 2 inputs are connected to 4 output units. Output lattice is 2-D in this network. Net input is computed for each output and the unit with maximum input is selected to be the winning output. Weight update rules compute neighbourhood of each neuron according to the Gaussian of Euclidean distance to the selected one.

2 Competitive Learning Network and Kohonen's Algorithm

In Kohonen's algorithm, an input pattern is presented to the feature mapping architecture (Figure 1). There are no lateral links among output neurons. Output units are fully-connected to input units via excitatory connections, w_{ij} denotes the strength from input ξ_j to the output O_i .

In competitive networks, the output neuron which attains maximum net input inhibits all other neurons

in the same layer. Thus, at one instant, only one *winner-take-all* unit may become active. If we place these units in a line or a plane as shown in Figure 1, with a "defined" neighbourhood constraint, non-activated units will be elicited by the winning unit due to the *quality* of neighbourhood relation. Thus, location of the winner unit will flow information to the nearby units, creating a topological structure. At the end of a training phase, similar input patterns should give rise to output units which are geometrically close to each other. To create such a behavior, weight update rules should be defined. However, first of all basic structure of the network should be specified.

Figure 1 is a sample network of 2-dimensional architecture, an input pattern is presented to the corresponding feature map. There are no lateral links among output neurons. Output units are fully-connected to input units via excitatory connections, w_{ij} denotes the strength from input ξ_j to the output O_i . The net input of an output unit is computed simply by summing up all input activities:

$$h_i = \sum_j w_{ij} \xi_j$$

For the input vector ξ ,

$$h_i = \mathbf{w}_i \cdot \xi$$

Since, the objective is to find the output unit with the maximum net input, winning unit i^* is:

$$\mathbf{w}_i \cdot \xi \leq \mathbf{w}_{i^*} \cdot \xi \quad (1)$$

If the weights are normalized, Equation 1 takes the form:

$$|\mathbf{w}_i - \xi| \geq |\mathbf{w}_{i^*} - \xi| \quad (2)$$

Equation 2 addresses the fact that, winner should be the weight vector which is topologically closest to the input vector.

Up to now, the structure of the feature mapping architecture and selection of winner-take-all unit is discussed. To obtain a feature extracting behavior, network should be trained with sufficient number of input patterns first, and connection weights should be modified according to the training set. Attention to

the close relation of weights with training set, weights are adjusted in order to extract the features of specific a training set. Algorithms works in the following manner: First weights are initiated randomly, different from each other enough to avoid symmetry breaking problem which occurs when weights are same initially. Then continually input patterns, ξ_j are selected randomly from training set. For each input pattern, a winning output unit i^* , which is topologically closest to the input vector is found by Equation 2. Then, weights of surrounding units as well as winning unit will be updated according to a distance criterion:

$$\Delta w_{ij} = \eta \Lambda(i, i^*) (\xi_j - w_{ij}) \quad (3)$$

for all i, j . η is the learning rate coefficient which specifies the rate of change in one step. For small η , it will converge very slow while for large η , it will adjust weights faster but with a higher probability of missing optimum value because step size is large. The neighbourhood function is selected as the Gaussian of the Euclidean distance between coordinates of output units:

$$\Lambda(i, i^*) = \exp\left(-\frac{|r_i - r_{i^*}|^2}{2\sigma^2}\right) \quad (4)$$

where $|r_i - r_{i^*}|$ is the Euclidean distance and σ is the parameter of Gaussian which defines the extent of neighbourhood. For large σ , effect to distant units is stronger and for very small σ , winning output could adjust only w_{i^*j} , itself.

Since all the algorithm heavily depends on the topological coordinates of output units, neighbourhood function, thus σ has crucial importance. Both parameters can be set to a static value initially and not changed, or they can be dynamically changed during learning. In the next section, both approaches are tested in similar tasks and optimum strategy is found at the end.

3 Experiments

Experiments are conducted using mainly three different data sets, Figures 3, 3 and 3 shows training sets which differ only in distribution in input space. In first set, 1000 inputs are randomly distributed in

the range of $[0, 1]$. The second data set differs from first one because density of data points in the inner square is twice as much of the rest, there are 1250 input points. In the third set, there are 750 data points which are randomly distributed in the range of $[0, 1]$ again, but in this case no data locate in a point with both coordinates are in the $[0.25, 0.75]$ range.

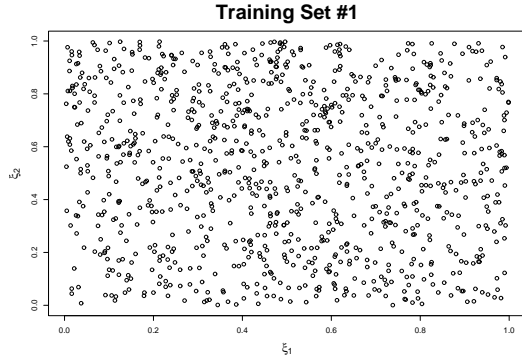


Figure 2: 1000 inputs randomly chosen in the range $\xi_1 = [0, 1]$ and $\xi_2 = [0, 1]$.

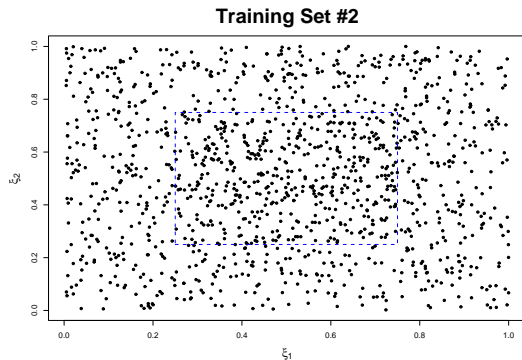


Figure 3: 1000 inputs randomly chosen in the range $\xi_1 = [0, 1]$ and $\xi_2 = [0, 1]$ and 250 more in the range $[0.25, 0.75]$.

In our simulations, two inputs described in the former paragraph will be mapped a 2-D grid, with 10×10 output units. The quality of the obtained results will be measured by the examining of Kohonen feature mapping [2], a sample initial mapping is shown

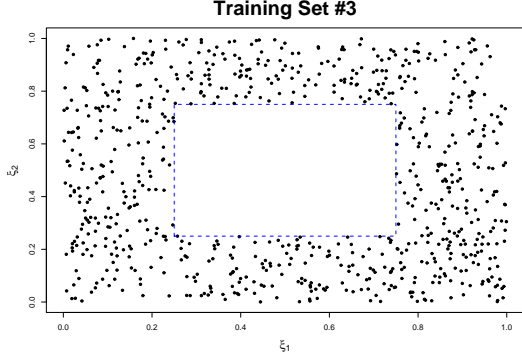


Figure 4: 750 inputs randomly chosen in the range $\xi_1 = [0, 1]$ and $\xi_2 = [0, 1]$ but it is not possible both inputs are in the range $[0.25, 0.75]$

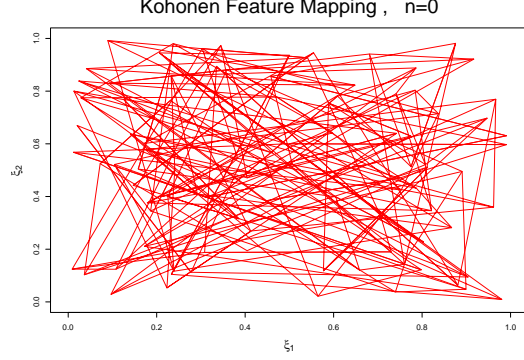


Figure 5: Initial state of the Kohonen feature mapping from a 2-D square region onto a 10x10 array of output units

in Figure 3. To construct this structure, each output unit is plotted as points with coordinates determined by their weights, in this case by two connection weights to two different inputs. Then according to the placement of output neurons on the 2-D lattice, adjacent points are connected with links in which 4-neighbourhood is taken as base criterion. As a result, topology of output array can be demonstrated as a *elastic net* structure in input space, which tries to come closer and closer during training phase [3]. In the result of this section, using above training set with different distributions, some simulations are runned and the effect of parameters on the behavior of the network will be analyzed.

Simulation 1

In this simulation, training set #1 which is shown in Figure 3 is used. The aim of this experiments is to analyze the effect of the two very important parameters, gain parameter (σ) of Gaussian from Equation 4 which determines the extent of neighbourhood and the coefficient (η) which specifies the step width in learning from Equation 3. Tens of experiments are done in order to optimize the parameters and obtain a topological correct feature map. However Kohonen's algorithm was unsuccessful in reaching a stable state with static parameters.

The learning rate parameter has a crucial effect on

the learning and properly adjusted initially. If learning coefficient is initialized to a small value $\eta = 0.2$, with a gain factor of $\sigma = 1$, network is proven to be unsuccessful to create good mappings. The behavior of the network highly depends on the initial configuration of weight vectors. Figure 3 demonstrates a situation where elastic net is folded initially and after a huge training period, no change in the net structure is obtained.

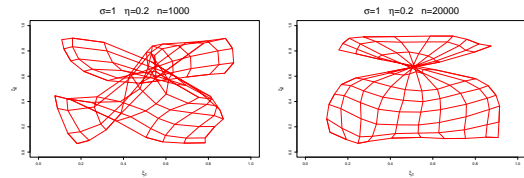


Figure 6: Kohonen feature mapping for Training Set #1, $\eta = 0.2$ and $\sigma = 1$. This figure illustrates that Self-Organizing Map is captured in a folded state with a small and static learning rate. Snapshots of network is taken in different steps, first is taken after iteration 1000 and second after iteration 20000.

Gain parameter of Gaussian should not be blamed for the results obtained. The gain is increased to $\sigma = 8$, with $\eta = 0.5$, and simulation results are examined during training process. Because extent of

neighbourhood is so wide, winning unit can affect other units in each learning phase in great extent such that, other points are not stretched enough apart one from other. Figure 3 shows this fact, while elastic net has a nice structure with correct weight values for adjacent output units, no matter the number of learning iterations, points are not distributed to cover input space and correctly represent the input samples.

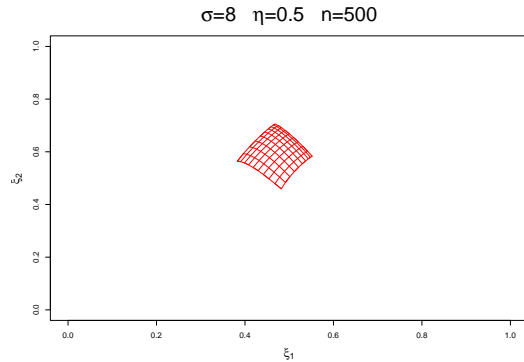


Figure 7: Kohonen feature mapping for Training Set #1, $\eta = 0.5$ and $\sigma = 8$. A full-covering Self-Organizing Map is not constructed because of the high gain parameter.

After tens of simulations, I found the optimum static parameter set to be $\eta = 1$ and $\sigma = 1$. In this configuration, while winning units can only update its own connection weights maximum and has little effect on other weights, learning rate ensures that the modifications give rise to the escaping network from some sticking conditions like folding of elastic net. Figure 3 shows a sample state of network for iteration $n=4000$. The elastic net is not well-structured because of the relatively high-learning rate. Additionally since learning rate is high, and only winning units' weights are adjusted, it was impossible to find a stable state after huge number of iterations, which is a sampling proof of the Kohonen's own words on the necessity of time-dependent adjustment of parameters 2. In next simulations, parameters will be adjusted dynamically during learning.

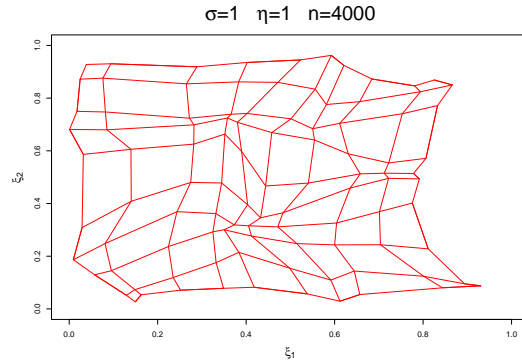


Figure 8: Kohonen feature mapping for Training Set #1, $\eta = 1$ and $\sigma = 1$. A non-stable and not well-structure mapping obtained due to the high learning rate.

Simulation 2

In [3], time dependence of ηt and σt is said to take various forms, in this simulation $a - bt$ form is used because it is easy to tune the a and b parameters. The main point is to enable the gain and learning rate gradually decrease in time and ensure that there will be a minimum for each of them. For learning rate:

$$\eta(t) = \begin{cases} a - bt & \text{if } \eta(t-1) > 0.05 \\ 0.05 & \text{ow.} \end{cases} \quad (5)$$

where $a = 1$, $b = 0.08$ and t , time is incremented for each 100 iterations. The gain of Gaussian is adjusted using following formula:

$$\sigma(t) = \begin{cases} a - bt & \text{if } \sigma(t-1) > 1 \\ 1 & \text{ow.} \end{cases} \quad (6)$$

where $a = 10$ and $b = 0.5$. Time-dependence of both parameters enabled the network to find optimum solutions. Figures 3,3 and 3 shows the state of self-organizing map in different steps of learning. In the iteration $n=200$, the extent of neighbourhood is wide so winning-units leads surrounding weights towards its own input weight location in the space as in Figure 3. However this time, both gain and learning rate is decreased gradually to a certain point while weight vectors covering the input space. Iterations after some point results in the learning of details of input data

set due to the small learning rate. As a result, at the end a well-structured nearly full covering elastic network emerges. As it is addressed in [4], a border which is not filled is appeared with a width inversely proportional to the linear size of output array.

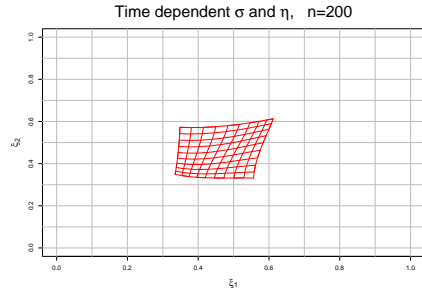


Figure 9: Kohonen feature map, elastic net is small because of the high σ , $n=100$.

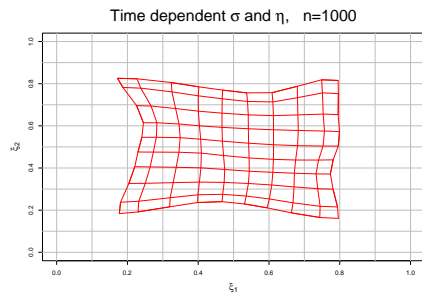


Figure 10: Kohonen feature map, elastic net started to stretch itself, $n=1000$.

Simulation 3

In this simulation, second training data set is used which is showed in Figure 3. The parameter adjustment which has a nature depend on time is established using Equations 5 and 6. The only difference from Simulation 2 is the distribution of data set in input space (you can find details of data set in Section 3). Figure 3 shows the state of self-organizing map after $n=2000$ iterations. The number of grids points of the network is larger where the probability is higher, since the distribution is non-uniform.

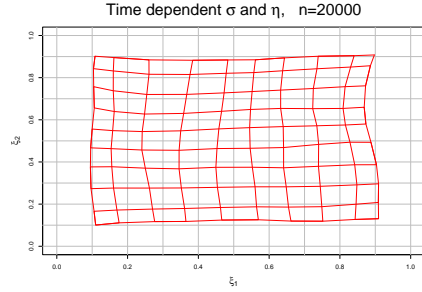


Figure 11: Well-structured Kohonen feature map, $n=20000$.

It proves that network correctly organize the feature map of training set #2.

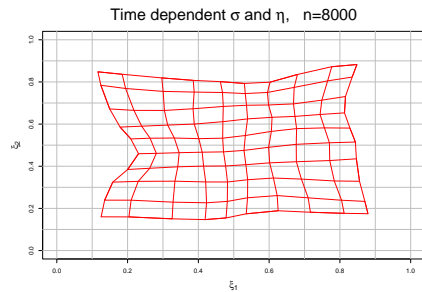


Figure 12: Kohonen feature map, for non-uniform training set with higher density inside (training set #2), after $n=8000$ iterations.

Simulation 4

Third data set (Figure 3) is used to obtain the reverse behavior of the Simulation 3. Training algorithm use Equations 5 and 6, to adjust weights dynamically. In this training set, there is no data point inside a rectangular regions, so the emerging feature map should contain less grid inside this region when compared to outer parts of the input space. Figure 3 shows the Kohonen feature map after 2000 iterations. As it is proposed lower probability leads a lower density of point.

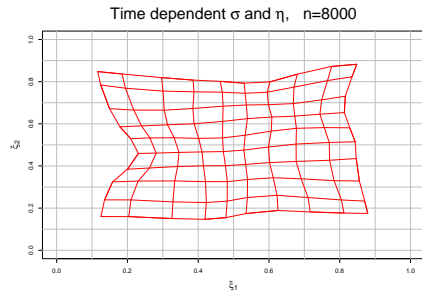


Figure 13: Kohonen feature map, for non-uniform training set with higher density inside (training set #2), after $n=8000$ iterations.

- [4] Teuvo Kohonen, Self-organization and Associative Memory. Berlin: Springer-Verlag, 1989.

4 Conclusion

In this work, Kohonen's self-organizing feature map's (SOMs) are analyzed in different training sets which are similar in data points' geometrical structure and different in data distribution on the input space. Emerging feature maps correctly represent the data sets. For uniformly distributed training sets, uniform and approximately full-covering mappings are obtained. In non-uniform sets, high probability leads high density of weight points. The other conclusion about SOMs is the importance of adjusting the parameters according to time. Both gain of Gaussian and learning rate coefficient should be gradually decreased to a certain point in order to obtain the best result.

References

- [1] E.R. Kandel, J.H. Schwartz, and T.M. Jessel, Appleton & Lange, Principles of Neural Science, Chapter 1, 1991.
- [2] Teuvo Kohonen, Self-organized formation of topologically correct feature maps, Biological Cybernetics 43: 59-69, 1982.
- [3] Hertz, Krogh, and Palmer, Introduction to the Theory of Neural Computation, Chapter 9, pp.217-250, Addison-Wesley, 1991.