Alexandria Engineering Journal (2014) 53, 827–831

HOSTED BY
ELSEVIER

Alexandria University

Alexandria Engineering Journal

www.elsevier.com/locate/aej
www.sciencedirect.com

ORIGINAL ARTICLE

A novel Self-Organizing Map (SOM) learning algorithm with nearest and farthest neurons

Vikas Chaudhary [a,*], R.S. Bhatia [a], Anil K. Ahlawat [b]

[a] *National Institute of Technology (N.I.T.), Kurukshetra, Haryana, India*
[b] *Krishna Institute of Engineering & Technology, Ghaziabad, U.P., India*

Received 3 May 2013; revised 11 September 2014; accepted 16 September 2014
Available online 23 October 2014

KEYWORDS

Self-Organizing Map (SOM);
Farthest neuron;
Nearest neuron;
Winning frequency;
Neighborhood neurons

Abstract The Self-Organizing Map (SOM) has applications like dimension reduction, data clustering, image analysis, and many others. In conventional SOM, the weights of the winner and its neighboring neurons are updated regardless of their distance from the input vector. In the proposed SOM, the farthest and nearest neurons from among the 1-neighborhood of the winner neuron, and also the winning frequency of each neuron are found out and taken into account while updating the weight. This new SOM is applied to various input data sets and the learning performance is evaluated using three standard measurements. It is confirmed that modified SOM obtained a far better result and better effective mapping as compared to the conventional SOM, which reflects the input data distribution.

1. Introduction

The Self-Organizing Map (SOM) is an unsupervised learning algorithm introduced by Kohonen [1]. In the area of artificial neural networks, the SOM is an excellent data-exploring tool as well [2]. It can project high-dimensional patterns onto a low-dimensional topology map. The SOM map consists of a one or two dimensional (2-D) grid of nodes. These nodes are also called neurons. Each neuron's weight vector has the same dimension as the input vector. The SOM obtains a statistical feature of the input data and is applied to a wide field of data classification [3–6]. SOM is based on competitive learning. In competitive learning [7], neuron activation is a function of distance between neuron weight and input data. An activated neuron learns the most and its weights are thus modified. If a similar pattern is found again, then the same neuron may be activated again. This means that a particular neuron wins repeatedly. So this neuron would learn more. To prevent this, conscience learning is a way, which had been proposed by De Sieno [8]. Further, Rival penalized competitive learning (RPCL) [9] and its variant Rival penalized controlled competitive learning (RPCCL) [10–13] was also proposed. SOM preserves the topology of input data by assigning each datum to a neuron having the highest similarity, and data with similar attributes are mapped into adjacent neurons [14].

The remainder of this paper is organized as follows. In Section 2, we explain the conventional SOM learning algorithm. In Section 3, the proposed SOM learning algorithm

* Corresponding author.
E-mail address: vikas_2610@rediffmail.com (V. Chaudhary).
Peer review under responsibility of Faculty of Engineering, Alexandria University.

http://dx.doi.org/10.1016/j.aej.2014.09.007
1110-0168

is explained. In Section 4, we conduct the experiments to compare the performance. In Section 5, we discuss the conclusion.

2. Self-Organizing Map (SOM)

The SOM consist of m neurons located at a regular low-dimensional map, usually a 2-D map. These neurons [15] are connected with their neighbors according to topological connections. There are two common types of topologies rectangular and hexagonal [16,17] for SOM map. Each neuron i has a d-dimensional weight vector w = ($(w_{i1}, w_{i2}, \ldots, w_{id})$, where $i = 1, 2, \ldots, m$, which has the same dimension as the input space.

The conventional SOM learning algorithm can be explained using the following steps:

(a) Initialize the weight vectors $w_i's$ of the $m \times n$ neurons.
(b) Randomly select an input vector $x(t)$ and it is input to all the neurons at the same time in parallel.
(c) Find the winner neuron c, i.e., BMU using the following equation:

$$c = \arg\left(\min_{1 \leqslant i \leqslant mn}\{\|w_i(t) - x(t)\|\}\right), \tag{1}$$

$||.||$ is the Euclidean distance measure. Where $x(t)$ and $w_i(t)$ are the input and weight vector of neuron i at iteration t respectively.

(d) The weight vector of the neurons is updated using the following equation:

$$w_i(t+1) = w_i(t) + h_{c,i}(t)[x(t) - w_i(t)], \tag{2}$$

where $h_{c,i}(t)$ is a Gaussian neighborhood function [16] given below:

$$h_{c,i}(t) = \alpha(t) \cdot \exp\left(-\frac{||r_c - r_i||^2}{2\sigma^2(t)}\right), \tag{3}$$

where r is the coordinate position of the neuron on the map, $\alpha(t)$ is the learning rate and $\sigma(t)$ is the width of neighborhood radius. Both $\alpha(t)$ and $\sigma(t)$ decrease monotonically using the following equation:

$$\alpha(t) = \alpha(0)\left(\frac{\alpha(T)}{\alpha(0)}\right)^{t/T}, \tag{4}$$

$$\sigma(t) = \sigma(0)\left(\frac{\sigma(T)}{\sigma(0)}\right)^{t/T} \tag{5}$$

where T is the training length.

(e) For all the input data, steps (b) to (d) are repeated.

3. Modified SOM

For each input data, the neurons at minimum and maximum distance from among 1-neighborhood of the BMU are found out as shown in Fig. 1. These are then named nearest and farthest neuron for that particular input. The proposed learning algorithm of SOM can be summarized in the following steps:

(Step 1) All the weight vectors $w_i \in \mathcal{M}$ of $m \times n$ neurons are initialized, where $i = 1, 2, \ldots, mn$ and $\mathcal{M}$ is a set of $m * n$ weight vectors. Then the winning frequency $\eta_i = 0$ is initialized for all neurons and the connection value $C_{(i,j)} = 0$ is also initialized between each neuron.

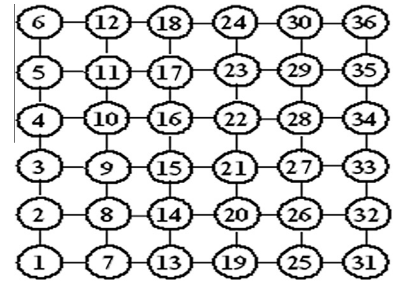

Figure 1 Neighborhood on the rectangular grid. Suppose $C = 16$, $Nc_1 = \{10, 15, 17, 22\}$. If $f = 15$, $S_f = \{13, 14\}$. If $f = 22$, $S_f = \{28, 34\}$.

(Step 2) An input vector $x(t)$ is selected randomly and given simultaneously to all the neurons.

(Step 3) The winner neuron c, i.e., BMU is found out using Eq. (1). Then, the distance between input $x(t)$ and weight vector is found and the rank $rank_i$ is assigned to each neuron, where $i = 0, 1, \ldots mn$. The rank $rank_i$ is taken to be 0 for the BMU, because of being nearest to the input vector. The winning frequency η_c of the winner neuron c is increased by 1.

(Step 4) The farthest neuron and the nearest neuron are found out from among the 1-neighborhood of BMU using Euclidean equation.

(Step 5) The connection value between BMU and neuron i is increased using the following equation:

$$C_{(c,i)} = C_{(c,i)} + 1, \tag{6}$$

where $i = f$ or $i \in S_f$.

Also, the relative winning frequency λ_i of the neuron i is calculated using the following equation:

$$\lambda_i = \eta_i \Big/ \sum_{j=1}^{\mathcal{M}} \eta_j \tag{7}$$

(Step 6) Except for the nearest neuron, the weight vectors of the winner neuron and its neighbors are updated using the following equation:

$$w_i(t+1) = w_i(t) + h_{c,i}(t)[x(t) - w_i(t)], \tag{8}$$

where the function $h_{c,i}(t)$ is the neighborhood function described as follows:

$$h_{c,i}(t) = \alpha(t) \cdot (1 - \lambda_i) \cdot \exp\left(-\frac{\gamma_{(c,i)}}{2\sigma^2(t)}\right), \tag{9}$$

$$\gamma_{(c,i)} = r_i + \left(||r_c - r_i||^2 + C_{(c,i)}\right), \tag{10}$$

Both $\alpha(t)$ and $\sigma(t)$ decrease consistently with time using Eqs. (4) and (5) respectively.

(Step 7) The weight vector of the nearest neuron is updated using the following equation:

$$w_q(t+1) = w_q(t) + h_{c,q}(t)[x(t) - w_q(t)], \tag{11}$$

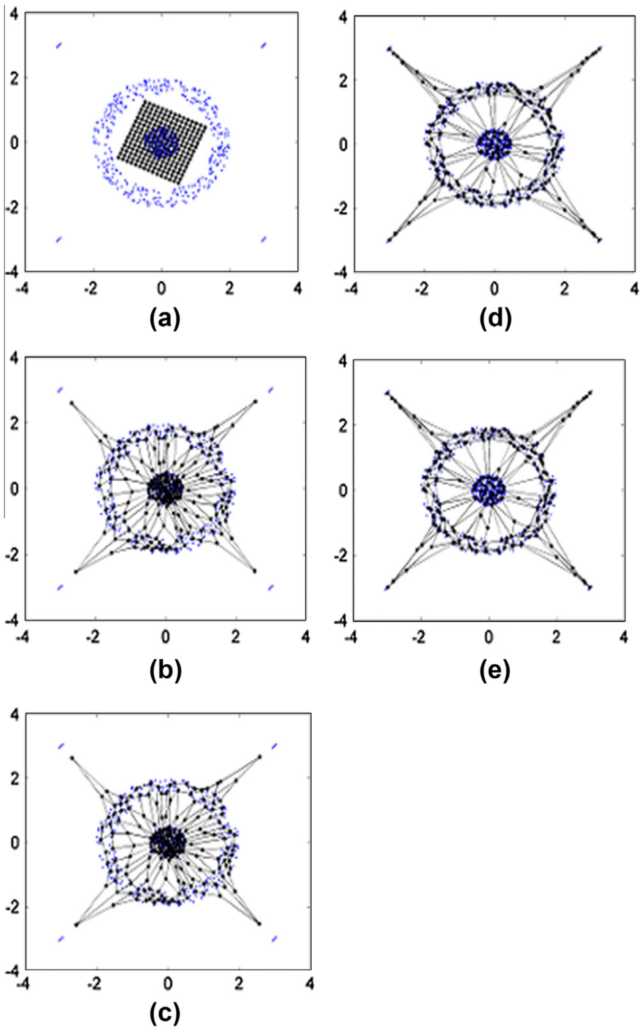

Figure 2 Snapshots of the trained map of the two algorithms using linear initialization in Experiment 1. (a) Linear initialized map, (b) conventional SOM (ten epochs), (c) conventional SOM (fifteen epochs), (d) modified SOM (ten epochs), (e) modified SOM (fifteen epochs).

Table 1 Quantization error Q_e, topographic error T_e and neuron utilization U for target dataset (Experiment 1).

Algorithm	Quality parameters		
	Q_e	T_e	U
Conventional SOM	0.0671	0.1922	0.99
Modified SOM	0.0598	0.0831	0.99

where the function $h_{c,q}(t)$ is the neighborhood function and described as follows:

$$h_{c,q}(t) = \alpha(t) \cdot (1 - \lambda_i) \cdot \exp\left(-\frac{\delta_{(c,q)}}{2\sigma^2(t)}\right), \tag{12}$$

$$\delta_{(c,q)} = ||w_q(t) - x(t)||^2 + C_{(c,q)}/m, \tag{13}$$

where $||w_q(t) - x(t)||$ is the distance between weight vector of nearest neuron and input vector $x(t)$.

(Step 8) The Steps 2 to7 are repeated for all the input data.

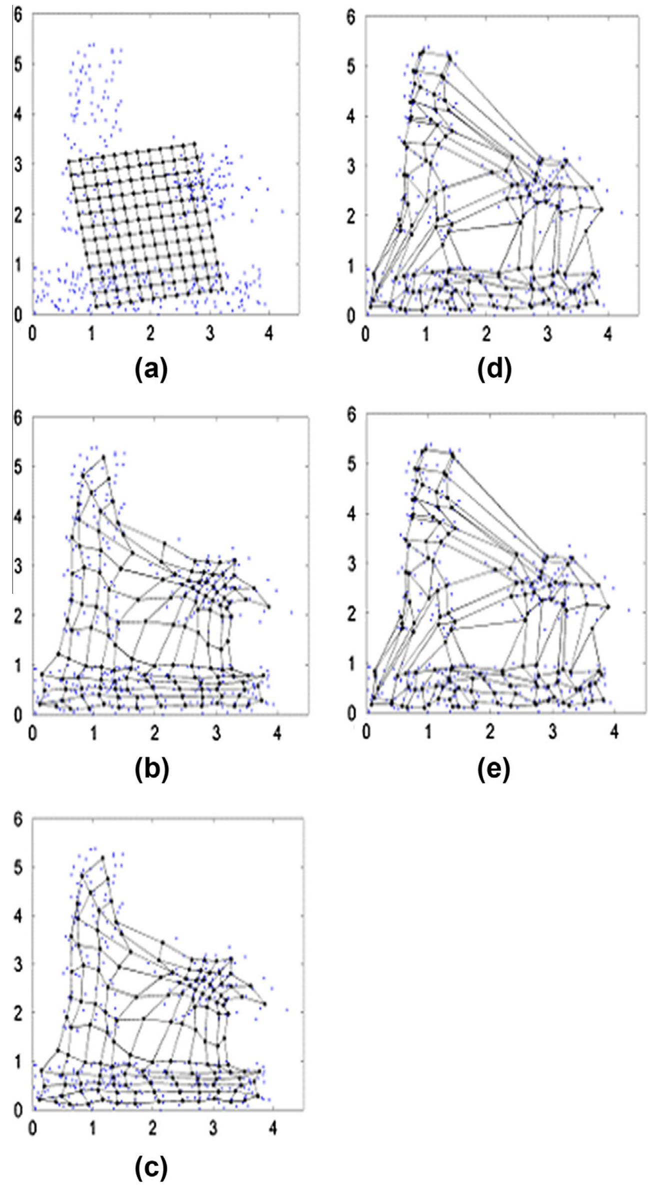

Figure 3 Snapshots of the trained map of the two algorithms using linear initialization in Experiment 2. (a) Linear initialized map, (b) conventional SOM (ten epochs), (c) conventional SOM (fifteen epochs), (d) modified SOM (ten epochs), (e) modified SOM (fifteen epochs).

4. Experimental results

Topographic Error, Quantization Error and Neuron Utilization [18–22] have been used to compare the learning performance of the modified SOM with conventional SOM.

4.1. Experiment 1

We carry out the learning experiment on Target dataset from UCI machine learning repository [23]. This dataset consists of 770 points and has an outliers clustering problem. We have taken 15×15 map size for both algorithms.

Table 2 Quantization error Q_e, topographic error T_e and neuron utilization U for LSun dataset (Experiment 2).

Algorithm	Quality parameters		
	Q_e	T_e	U
Conventional SOM	0.0962	0.1350	1.0
Modified SOM	0.0891	0.0525	1.0

The parameters for the learning are taken as given below:

(For SOM)

$$\alpha(0) = 0.9, \quad \alpha(0) = 0.001, \; \sigma(0) = 8.0, \quad \sigma(T) = 0.001$$

(For modified-SOM)

$$\alpha(0) = 0.9, \quad \alpha(0) = 0.001, \; \sigma(0) = 8.0, \quad \sigma(T) = 0.001$$

The experimental results of both algorithms using linear initialization are shown in Fig. 2. It can be observed that the modified SOM reaches to the outlier's input data. Also the modified SOM covers the input data more effectively as compared to the conventional SOM. From these figures, it can be concluded that the modified SOM obtains a more effective map, which is more organized in every corner of the input as compared to the conventional SOM. The topographic and quantization error of the modified SOM are smaller than the conventional SOM as shown in Table 1. From Table 1, it can be concluded that there is a good improvement of 56.76% and 10.88% in topographic and quantization error respectively.

4.2. Experiment 2

The modified SOM is applied on LSun dataset from UCI machine learning repository [23]. This dataset consists of 400 points, and has different variances and inter cluster distances. The map size is set to 12×12 in the conventional and modified SOM. The parameters for the learning are chosen as follows:

(For SOM)

$$\alpha(0) = 0.9, \quad \alpha(0) = 0.001, \; \sigma(0) = 6.0, \quad \sigma(T) = 0.001$$

(For modified-SOM)

$$\alpha(0) = 0.9, \quad \alpha(0) = 0.001, \; \sigma(0) = 6.0, \quad \sigma(T) = 0.001$$

The simulation results of both algorithms are shown in Fig. 3 using linear initialization. From these figures, it can be concluded that the modified SOM reaches better in every corner of the input data as compared to the conventional SOM. The topographic and quantization error of the modified SOM are also smaller than the conventional SOM as shown in Table 2. As can be seen from Table 2, an improvement of 61.11% and 7.38% was achieved in topographic and quantization error respectively.

5. Conclusion

We have proposed a modified learning algorithm of SOM, in which the farthest and nearest neurons are found from among the 1-neighborhood of the winner neuron. The learning performance is then calculated using three standard measurements. The modified SOM is applied to various standard input data sets, and the experimental results prove that the modified SOM is better than the conventional SOM. It is also shown that modified SOM reaches to the outlier's data, where conventional SOM could never reach. The results have shown that the modified SOM preserves the topology of input dataset more efficiently and perfectly. The modified SOM have much lower topographic and quantization error, which is desirable.

References

[1] T. Kohonen, Self-Organizing Maps, Springer, 1995.

[2] Shu-Ling Shieh, I-En Liao, A new approach for data clustering & visualization using self-organizing maps, Expert Syst. Appl. 39 (2012) 11924–11933.

[3] Y. Cheng, Clustering with competing self-organizing maps, in: Proc. of IJCNN, vol. IV, pp. 785–790, 1992.

[4] W. Wan, D. Fraser, M2dSOMAP: clustering and classification of remotely sensed imagery by combining multiple Kohonen self-organizing maps and associative memory, in: Proc. of IJCNN, vol. III, pp. 2464–2467, 1993.

[5] J. Vesanto, E. Alhoniemi, Clustering of the self-organizing map, IEEE Trans. Neural Networks 11 (3) (2002) 586–600.

[6] Lapidot, H. Guterman, A. Cohen, Unsupervised speaker recognition based on competition between self-organizing maps, IEEE Trans. Neural Networks 13 (4) (2002) 877–887.

[7] E. Rumelhart, D. Zipser, Feature discovery by competitive learning, Cognitive Sci. 9 (1985) 75–112.

[8] D. DeSieno, Adding a conscience to competitive learning, in: Proc. ICNN'88, International Conference on Neural Networks, Piscataway.

[9] L. Xu, A. Krzyzak, E. Oja, Rival penalized competitive learning for clustering analysis, RBF net, and curve detection, IEEE Trans. Neural Networks 4 (4) (1993) 636–649.

[10] Y.M. Cheung, Rival penalization controlled competitive learning for data clustering with unknown cluster number, in: Proc. 9th Int. Conf. Neural Inf. Process, Singapore, 2002.

[11] Y.M. Cheung, On rival penalization controlled competitive learning for clustering with automatic cluster number selection, IEEE Trans. Knowledge Data Eng. 17 (11) (2005) 1583–1588.

[12] H. Yin, Data visualization and manifold mapping using the ViSOM, IEEE Trans. Neural Networks 15 (2002) 1005–1016.

[13] Guénaël Cabanes, Younès Bennani, D. Fresneau, Enriched topological learning for cluster detection and visualization, Neural Networks 32 (2012) 186–195.

[14] T. Kohonen, Self-Organizing Maps, third ed., Springer-Verlag, Berlin, Heidelberg, New York, 2001.

[15] H. Yang, M. Palaniswami, On the issue of neighborhood in self-organizing maps, in: Proc. ACM/SIGAPP Symp. Appl. Comput., pp. 412–416, 1992.

[16] Y. Cheung, L. Law, Rival model penalized self organizing map, Trans. Neural Networks 18 (1) (2007) 289–295.

[17] Vikas Chaudhary, Anil K. Ahlawat, R.S. Bhatia, An efficient self-organizing map learning algorithm with winning frequency of neurons for clustering application, in: Proc. International Advanced Computing Conference (IACC), IEEE, pp. 664–668, 2013.

[18] Vikas Chaudhary, R.S. Bhatia, Anil K. Ahlawat, The self-organizing map learning algorithm with inactive and relative winning frequency of active neurons, HKIE Trans. 21 (1) (2014) 62–67.

[19] A. Ultsch, Clustering with SOM, in: Proc. of Workshop on Self-Organizing Features Maps, pp. 31–37, Paris, 2005.

[20] N.J. Mount, D. Weaver, Self-organizing maps and boundary effects: quantifying the benefits of torus wrapping for mapping SOM trajectories, Pattern Anal. Appl. 14 (2) (2011) 139–148.
[21] Joseph P. Herbert, Jing Tao Yao, A granular computing framework for self-organizing maps, Neurocomputing 9 (2009) 2865–2872.
[22] H. Matsushita, Y. Nishio, Batch-learning self-organizing map with weighted connections avoiding false-neighbor effects, IJCNN, IEEE (2010) 1075–1080.
[23] K. Bache, M. Lichman, UCI Machine Learning Repository, University of California, School of Information and Computer Science, Irvine, CA, 2013. <http://archive.ics.uci.edu/ml>.