International Journal of Computational Intelligence Systems, Vol. 7, No. 5 (October 2014), 963-972

An efficient self-organizing map (E-SOM) learning algorithm using group of neurons

Vikas Chaudhary
National Institute of Technology (N.I.T.),
Kurukshetra, Haryana, India
E-mail: ramvikaskec@yahoo.com

R.S. Bhatia
National Institute of Technology (N.I.T.),
Kurukshetra, Haryana, India

Anil K. Ahlawat
Krishna Institute of Engineering & Technology,
Ghaziabad, U.P., India

Received 8 May 2013

Accepted 1 September 2014

Abstract

In the learning process of the conventional SOM, the neuron which is closer to the winner neuron learns more than the neuron which is farther away from the winner neuron. The neurons farther away from input are not able to learn properly and some dead units are left on the map. To decrease dead unit problem and improve the learning efficiency, an efficient Self-organzing map algorithm using group of neurons has been proposed. In this paper, we have divided the neurons on the map into two groups according to distance from input: normal and distant. The neurons which are far away from the input have been named distant neurons. We have done some changes in the kernel function for the distant neurons and then compared the learning efficiency of the algorithms by applying on standard input dataset. The results have been compared using three well known parameters, which are widely accepted for checking the learning efficiency of machine learning algorithms. It has been observed from the experimental results that proposed SOM successfully decrease dead units,while still preserving the topology of input data with lesser errors. The maps achieved by the proposed SOM have a lower error measure than the maps formed by SOM and false neighbor degree SOM (FN-SOM).

Keywords: Self-organizing map (SOM); kernel function; distant neuron; Efficient SOM (E-SOM).

1. Introduction

A Self-organizing map (SOM) is an unsupervised learning algorithm[1] introduced by T. Kohonen. The SOM is a model of self-organizing process of the brain. A SOM map possesses a one or two dimensional (2-D) grid of nodes. Each node on the map can also be called a neuron. Each neuron on the map has a weight vector, which is of the same dimension as the input vector. A SOM obtains a statistical feature of the input data and is applied to a wide field of vector quantization, dimension reduction and data clustering[3-6]. It can project high-dimensional data onto a low-dimensional topology map.

The SOM is based on competitive learning. In competitive learning, neuron activation is a function of the distance between neuron weight and input data. An activated neuron learns the most and it's weight is modified. If a similar pattern is found again, then there are more chances that the same neuron may be activated. This results in the same neuron winning repeatedly. Thus this particular neuron learns more. To prevent this effect, conscience learning is a way, which

has been proposed by De Sieno[7]. The Rival penalized competitive learning (RPCL)[8] and Rival penalized controlled competitive learning (RPCCL)[9-10] were also proposed to prevent the same effect. Further, modifications in SOM have been proposed for better visualization[2,11,12]. A rival-model penalized self organizing map (RPSOM) learning algorithm[13] inspired by the idea of the rival penalized competitive learning (RPCL) and rival penalization controlled competitive learning (RPCCL) approach was proposed. For each input, the RPSOM chooses several rivals of the best-matching unit (BMU) and penalizes their associated models a little far away from the input. Then an updated SOM with false neighbor degree (FN-SOM)[14] was proposed. This FN-SOM worked on the false-neighbor of the neuron, which wins the least. The FN-SOM allocates false-neighbor degree between adjacent rows and columns. It increases the learning efficiency, but the neurons on same line increase simultaneously as an after-effect. This often produces the increase in FNDs between neurons having real neighbors as a side-effect. It often produces the twist of the map[15]. A Weighted Connections SOM (WC-SOM)[15] was proposed. In WC-SOM, all the connections between adjacent neurons are weighted to avoid false-neighbor effects unlike FN-SOM. These weights are known as false-neighbor weights (FNWs). WC-SOM changes the neighborhood relationship more flexibly according to the situation and the shape of data. But false-neighbor weights (FNWs) act as a burden and increase the complexity. A fast training SOM (ftSOM) method was proposed[16]. In ftSOM the steps of conventional SOM are rearranged and the learning speed is increased. A modification in SOM model was proposed[17], where some BMUs were excluded so that they don't affect the others and become refractory for the rest of the process. Some other variants have been proposed to keep neighborhood aside or redefined[18,19,20]. A modified SOM was proposed to solve the problems related to the initialization of neurons. This approach used the Hilbert curve to initialize neurons and converged in less epochs[21].

In SOM, neurons learn according to the distance from the winner on the map. The neurons farther away from the input are not able to learn properly and remain as dead units on the map. As a result, a degradation in the learning efficiency of SOM is noted. To decrease dead units and increase learning efficiency of SOM, we divide the neurons on the map into two groups according to the distance from the input: normal and distant as shown in Fig. 1. We have considered twenty percent of the total neurons on the map as distant neurons according to a study[22]. Some changes in the kernel function for the distant neurons is done. The learning efficiency of E-SOM was compared with conventional SOM and FN-SOM by applying on standard input dataset. It is observed through the simulation results that E-SOM reduced dead units and enhanced the learning capabilities with lesser errors.

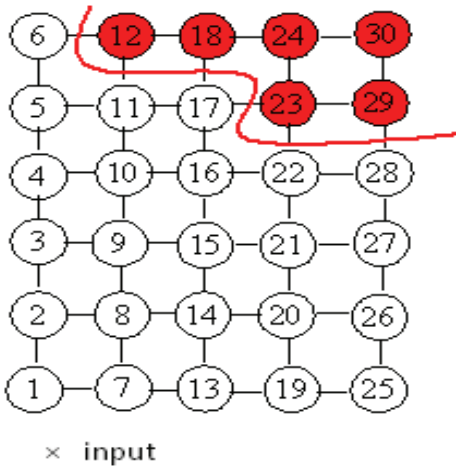

Fig. 1.Two groups of neurons on the map (6×5) are separated by red line: The distant neurons are $\{12, 18, 23, 24, 29, 30\}$ colored red. Rest neurons are normal neurons. × indicates the input data.

The paper has been divided into 5 sections. In Section 2, we explain the SOM learning algorithm and E-SOM learning algorithm in section 3. In Section 4, we conduct the experiments to compare the performance of our approach with SOM and FN-SOM, followed by conclusion in Section 5.

2. Self-Organizing Map (SOM)

The SOM contains a set of artificial neurons, each with a weight vector[23]. In SOM, m neurons are located at a low-dimensional map, generally a 2-D map. For example, $m \times n$ neurons are connected with their neighbors to form a map. These neurons are connected with their neighbors according to topological connections.There are two common types of topologies for SOM map: rectangular and hexagonal[24-25]. In rectangular topology, a neuron has four one-neighbor neurons and in hexagonal, a neuron has six one-neighbor neurons as shown in Fig. 2. Each neuron i has weight vector

$w = (w_{i1}, w_{i2}, \ldots\ldots, w_{id})$, where $i = 1, 2 \ldots\ldots m$, which has the same dimension as the input dataset. In each iteration, an input $x(t)$is randomly selected from the input dataset, and the winner neuron, which is also called the best matching unit (BMU), is found. The SOM is based on competitive learning, so all neurons on the map compete with each other to become the winner. The neuron with the least Euclidean distance between its weight vector and the input will be the BMU neuron. The learning algorithm of SOM is explained in the next subsection.

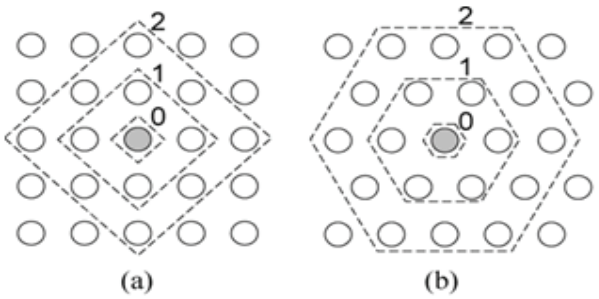

Fig. 2. Two commonly used topologies in SOM.
(a) Rectangular. (b) Hexagonal.

2.1. *Learning Algorithm*

The conventional SOM learning algorithm can be explained as follows:

Step 1: The weight vectors $w_i's$ of all the neurons on the map (i.e., $m \times n$ neurons) are initialized. The size of weight vector w has to be same as input vector.

Step 2: An input vector $x(t)$ is randomly selected from the input dataset and given to all the neurons on the map.

Step 3: The winner neuron (BMU) is found and its index c is stored using the following equation:

$$c = \arg(\min_{1 \le i \le mn} \{\| w_i(t) - x(t) \|\}), \quad (1)$$

where $\|.\|$ is the Euclidean distance measure and $w_i(t)$ is the weight vector of the neuron i at iteration t.

Step 4: The weight vectors of the neurons are updated using the following equation:

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

where t is the learning step and c is the index of BMU. $h_{c,i}(t)$is the Gaussian neighborhood function, which is widely used as a neighborhood function[24].The value of neighborhood function is maximum for the winner neuron and decreases with increasing distance from the BMU.

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

where $\|r_c - r_i\|$ is the distance between winner neuron c and neuron i on the map; and $\alpha(t)$ and $\sigma(t)$ are called the learning rate and width of neighborhood radius respectively. After each iteration t, $\alpha(t)$ and $\sigma(t)$ decrease using the following equation:

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

and

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

where T is the training length, α (0) is the initial learning rate and $\sigma(0)$ is the initial neighborhood radius.

Step 5: These steps are repeated for all the input data.

3. Efficient Self-Organizing Map (E-SOM)

To decrease the dead units and increase the learning efficiency, an efficient SOM (E-SOM) learning algorithm has been proposed using group of neurons on the map. In this approach, distant and normal neurons on the map for each input are identified. So we have made two groups of neurons on the map normal and distant.

The details of the E-SOM learning algorithm are explained as follows:

Step 1: The weight vectors w_i of $m \times n$ neurons, where *i= 1, 2,...., mn*, are initialized.

Step 2: An input vector $x(t)$ is randomly selected and it is input to all the neurons at the same time in parallel.

Step 3: The winner neuron *c*, i.e., BMU is found using "Eq. (1)".

Step 4: The distance of all neurons from the input $x(t)$ is calculated using Euclidean formula, and 20% of these, which are most distant from this input are identified. These are the distant neurons of this input. In Fig. 1, a map of size of 6 × 5, the six neurons in red color denote the distant neurons for the input vector $x(t)$.

Step 5: The weight vectors of winner and its neighbors except the distant neurons are updated using "Eq. (2)".
Step 6: The weight vectors of distant neurons are updated using the following equation:

$$w_k(t+1) = w_k(t) + \alpha_d(t).h_{c,k}(t)[x(t) - w_k(t)], \quad (6)$$

where $\alpha_d(t)$ is the learning rate of the distant neurons, $0 < \alpha_d(t) < 0.1$. It decreases monotonically with time using "Eq. (4)". The function $h_{c,k}(t)$ is the neighborhood function for distant neurons and is described using the following equation:

$$h_{c,k}(t) = \delta.\exp\left(-\frac{\|r_c - r_k\|^2}{2\sigma^2(t)}\right), \quad (7)$$

$$\delta = \exp\left(-\frac{\|w_c(t) - x(t)\|^2}{2\sigma^2(t)}\right), \quad (8)$$

where $\|.\|$ is the Euclidean distance measurement.
Step 7: The Steps 2 to 6 are repeated for all the input data.

4. Experimental Results

To compare the learning performance of the proposed SOM with SOM and FN-SOM, we have used three standard datasets from UCI machine learning repository[27] and three quality parameters are used.

4.1. *Quality Criteria*

The following three well-known quality criterions, which are widely accepted for comparing the learning efficiency of machine learning algorithms, have been used:

(i) Quantization Error (Q_e): It measures the average distance from each input data to its winner[13].

$$Q_e = \frac{1}{N}\sum_{i=1}^{N} \| x_i - \overline{w}_i \|, \quad (9)$$

where N is total number of input data, $\overline{w}_i$ is the weight vector of the corresponding winner for the input vector x_i, and $\|.\|$ denotes the Euclidean distance. So Q_e should be small.

(ii) Topographic Error(*Te*): This describes how well the SOM preserves the topology of the input dataset[26].

$$T_e = \sum_{j=1}^{N} u(x_j) \Big/ N, \quad (10)$$

where N is the total number of input data. $u(x_j) = 1$ if the winner and the 2nd winner are not 1-neighbors, otherwise $u(x_j) = 0$. A smaller value of T_e means a better learning efficiency.

(iii) Neuron Utilization (*U*): It measures the percentage of neurons on the map that have become winners for the input data for one or more times[13].

$$U = \frac{1}{mn}\sum_{i=1}^{mn} u_i, \quad (11)$$

where $u_i = 1$ if the neuron i has won once or more; otherwise $u_i = 0$. U = 1.0 means all the neurons become winner. So U nearer to 1.0 is required.

4.2. *Experiments on Standard Datasets*

The following three well-known standard input datasets from UCI machine learning repository[27] have been used to compare the results:

A. Experiment 1

A comparison on learning capabilities has been done using the Target dataset from UCI machine learning repository[27]. This input dataset has 770 points and has an outlier's problem. The experiment is done using different parameters as shown in Table 1. It is not possible to visualize all results. The simulation results for map size = 12*15, $\alpha(0) = 0.7$, $\alpha_a = 0.5$, $\alpha_d = 0.03$, Epoch = 20 has been shown in Fig. 3.

It can be observed from Fig. 3 that neurons learn the input distribution more precisely using E-SOM approach. Through our approach, the number of neurons inside the circle has decreased, thus decreasing the dead units. As can be seen, more neurons are now reaching closer to the outlier's data on all four corners.
From Table 1, it can be concluded that the E-SOM preserves the topology of input data in a better manner and increases the neuron utilization also. The smaller value of topographic error in comparison to SOM and FN-SOM thus indicates that topology of input data is better preserved in the output map using E-SOM. The results indicate that E-SOM learns the input data with less topographic and quantization error in comparison to SOM and FN-SOM.

B. Experiment 2

A comparison on the learning capabilities is carried out on TwoDiamonds dataset from UCI machine learning

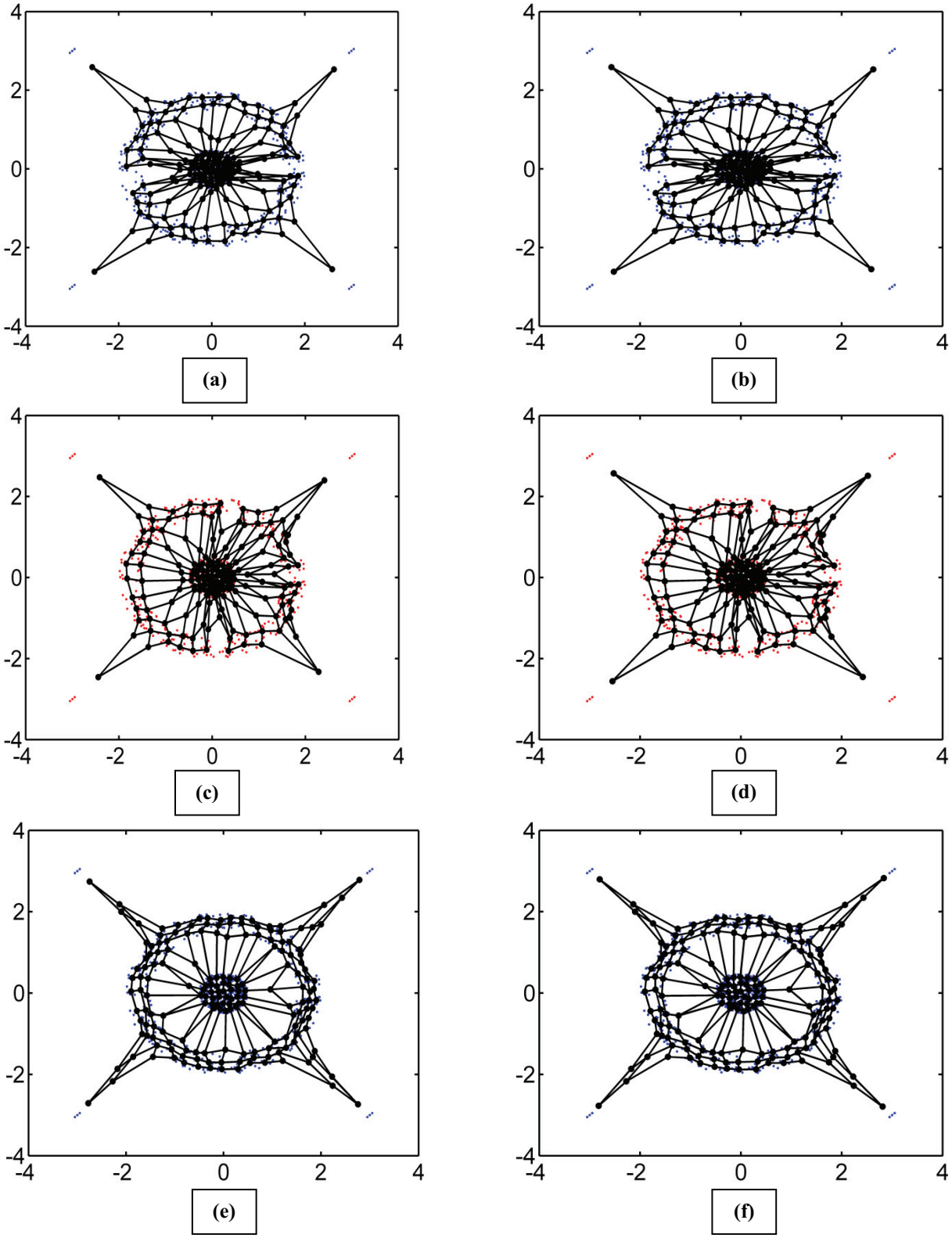

Fig. 3. Snapshots of the trained map of SOM, FN-SOM and E-SOM using linear initialization for map size 12*15 in Experiment 1. (a) Conventional SOM (ten epochs) (b) Conventional SOM (twenty epochs) (c) FN-SOM (ten epochs) (d) FN-SOM (twenty epochs) (e) E-SOM (ten epochs) (f) E-SOM (twenty epochs).

Table 1. Quantization error Qe, Topographic error Te and Neuron Utilization U for the Target dataset (Experiment 1).

Learning Parameters	Algorithm	Quality Parameters		
		QE	TE	U
Map =6*5, α(0)=0.5, $\alpha_a = 0.4$, $\alpha_d = 0.03$, Epoch=10	SOM	0.2018	0.2481	1.0
	FN-SOM	0.2010	0.3896	1.0
	E-SOM	0.1988	0.2688	1.0
Map =10*10, α(0)=0.6, $\alpha_a = 0.4$, $\alpha_d = 0.04$, Epoch=15	SOM	0.1008	0.1558	1.0
	FN-SOM	0.1002	0.2182	1.0
	E-SOM	0.0944	0.1405	1.0
Map =12*15, α(0)=0.7, $\alpha_a = 0.5$, $\alpha_d = 0.03$, Epoch=20	SOM	0.0711	0.1364	1.0
	FN-SOM	0.0701	0.1217	1.0
	E-SOM	0.0660	0.1156	1.0
Map =15*17, α(0)=0.4, $\alpha_a = 0.4$, $\alpha_d = 0.04$, Epoch=25	SOM	0.0635	0.1390	1.0
	FN-SOM	0.0601	0.1312	1.0
	E-SOM	0.0523	0.1091	1.0
Map =20*18, α(0)=0.6, $\alpha_a = 0.5$, $\alpha_d = 0.04$, Epoch=30	SOM	0.0451	0.1377	1.0
	FN-SOM	0.0412	0.1571	1.0
	E-SOM	0.0356	0.1195	1.0

Table 2. Quantization error Qe, Topographic error Te and Neuron Utilization U for the 2-Diamonds dataset (Experiment 2)

Learning Parameters	Algorithm	Quality Parameters		
		QE	TE	U
Map =7*5, α(0)=0.6, $\alpha_a = 0.3$, $\alpha_{in} = 0.02$, Epoch=10	SOM	0.1378	0.2737	1.0
	FN-SOM	0.1323	0.2550	1.0
	E-SOM	0.1201	0.1950	1.0
Map =10*10, α(0)=0.5, $\alpha_a = 0.4$, $\alpha_{in} = 0.03$, Epoch=15	SOM	0.0805	0.2275	1.0
	FN-SOM	0.0797	0.2450	1.0
	E-SOM	0.0761	0.2111	1.0
Map =15*14, α(0)=0.8, $\alpha_a = 0.5$, $\alpha_{in} = 0.04$, Epoch=20	SOM	0.0498	0.2325	1.0
	FN-SOM	0.0413	0.2387	1.0
	E-SOM	0.0387	0.2201	1.0
Map =18*15, α(0)=0.6, $\alpha_a = 0.5$, $\alpha_{in} = 0.03$, Epoch=25	SOM	0.0435	0.1925	1.0
	FN-SOM	0.0435	0.2288	1.0
	E-SOM	0.0412	0.1872	1.0
Map =25*20, α(0)=0.7, $\alpha_a = 0.4$, $\alpha_{in} = 0.04$, Epoch=30	SOM	0.0296	0.2450	1.0
	FN-SOM	0.0274	0.2575	0.9960
	E-SOM	0.0274	0.2362	1.0

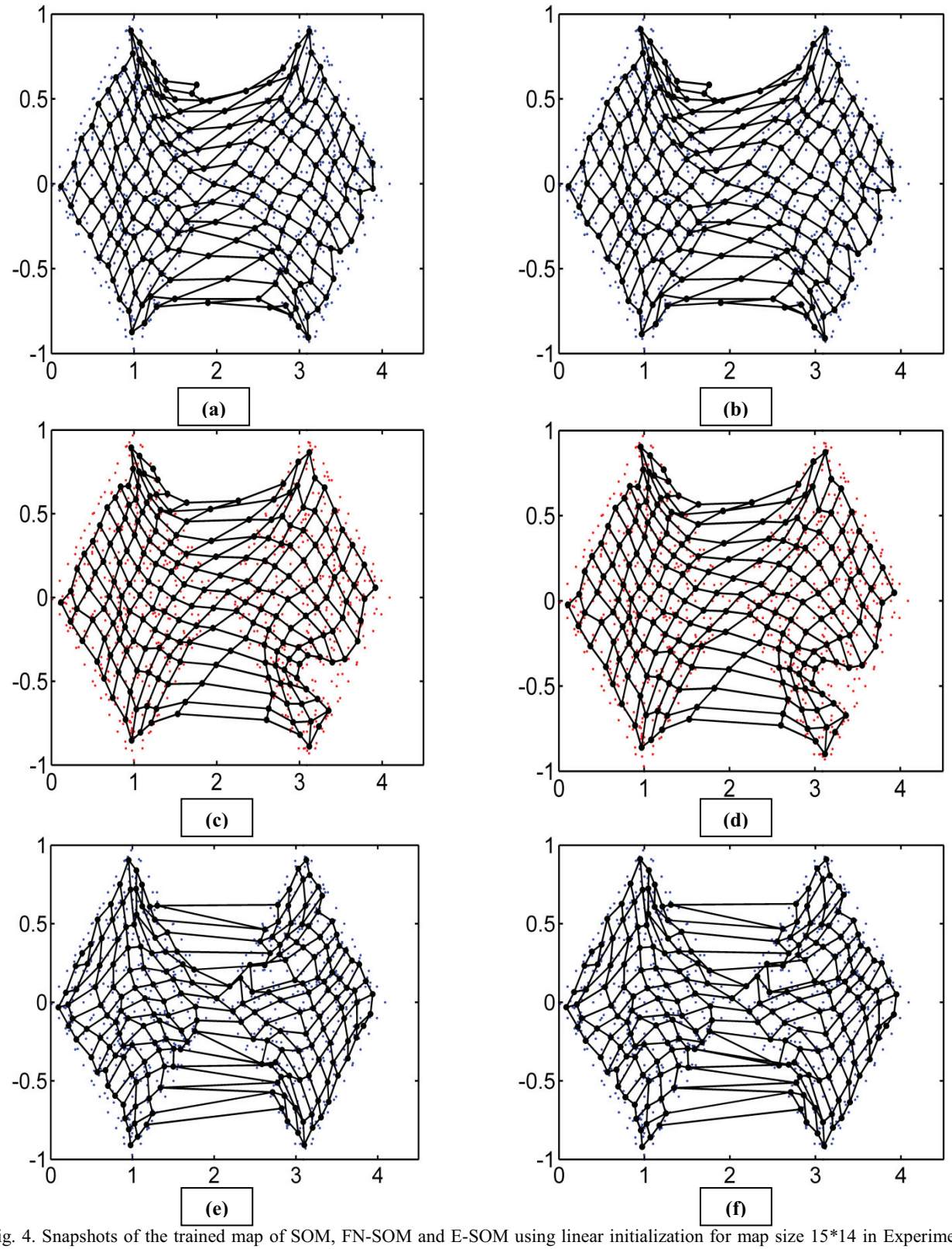

Fig. 4. Snapshots of the trained map of SOM, FN-SOM and E-SOM using linear initialization for map size 15*14 in Experiment 2. (a) Conventional SOM (ten epochs) (b) Conventional SOM (twenty epochs) (c) FN-SOM (ten epochs) (d) FN-SOM (twenty epochs) (e) Efficient SOM (ten epochs) (f) Efficient SOM (twenty epochs).

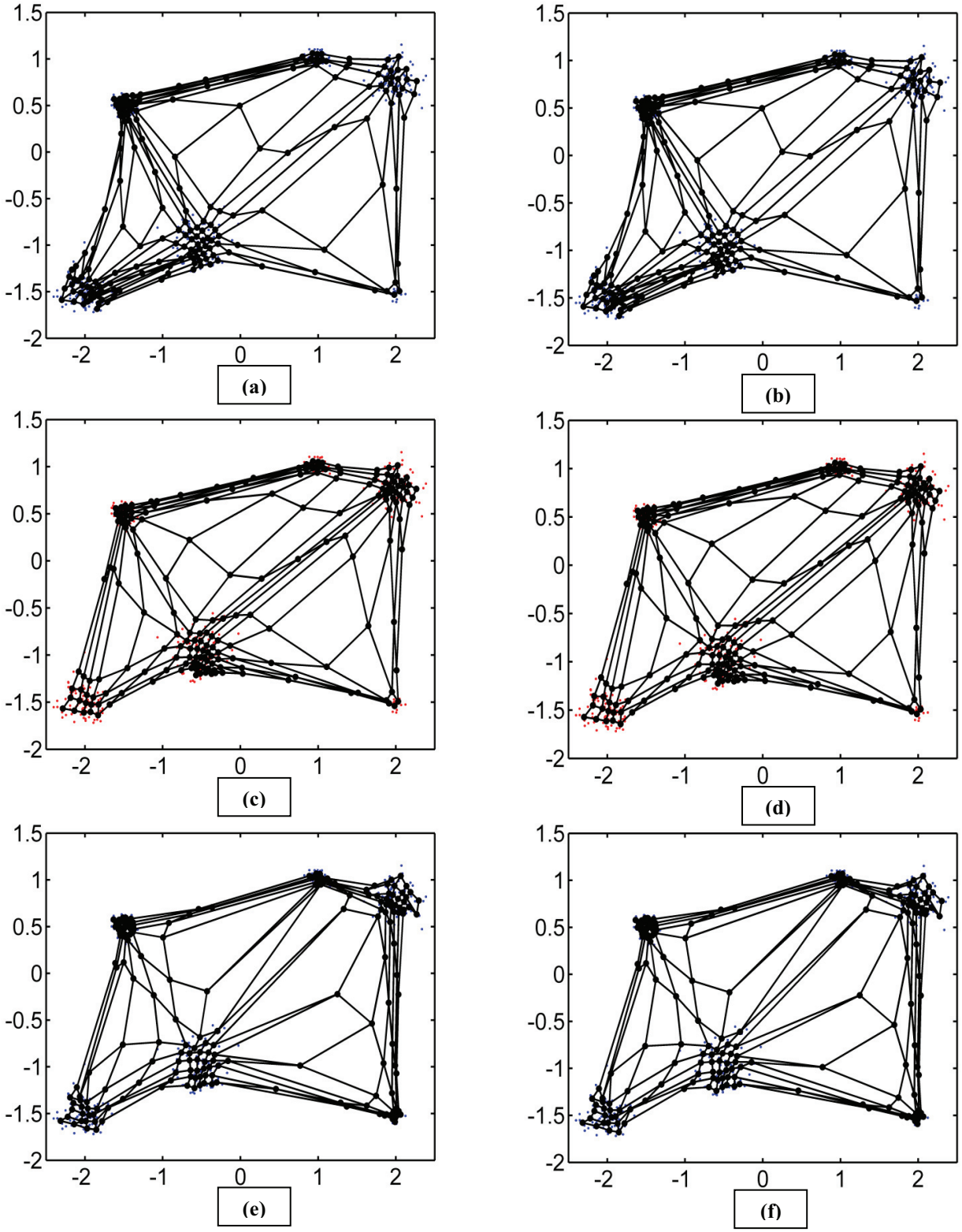

Fig. 5. Snapshots of the trained map of SOM, FN-SOM and E-SOM using linear initialization for map size 13*15 in Experiment 3. (a) Conventional SOM (ten epochs) (b) Conventional SOM (twenty epochs) (c) FN-SOM (ten epochs) (d) FN-SOM (twenty epochs) (e) E-SOM (ten epochs) (f) E-SOM (twenty epochs).

Table 3. Quantization error Qe, Topographic error Te and Neuron Utilization U for the Gaussian Mixture dataset (Experiment 3).

Learning Parameters	Algorithm	Quality Parameters		
		QE	TE	U
Map =6*7, α(0)=0.7, $\alpha_a = 0.3$, $\alpha_d = 0.02$, Epoch=15	SOM	0.0756	0.1553	1.0
	FN-SOM	0.0743	0.1809	1.0
	E-SOM	0.0629	0.1447	1.0
Map =11*10, α(0)=0.6, $\alpha_a = 0.4$, $\alpha_d = 0.03$, Epoch=18	SOM	0.0435	0.1851	1.0
	FN-SOM	0.0409	0.2072	1.0
	E-SOM	0.0368	0.1723	1.0
Map =13*15, α(0)=0.9, $\alpha_a = 0.4$, $\alpha_d = 0.04$, Epoch=20	SOM	0.0308	0.2660	0.9846
	FN-SOM	0.0287	0.2128	0.9846
	E-SOM	0.0205	0.1596	1.0
Map =17*16, α(0)=0.5, $\alpha_a = 0.5$, $\alpha_d = 0.03$, Epoch=24	SOM	0.0250	0.1426	0.9926
	FN-SOM	0.0251	0.2191	0.9853
	E-SOM	0.0162	0.1191	1.0
Map =22*25, α(0)=0.8, $\alpha_a = 0.3$, $\alpha_d = 0.04$, Epoch=35	SOM	0.0147	0.2383	0.9673
	FN-SOM	0.0138	0.2277	0.9709
	E-SOM	0.0124	0.1681	0.9846

repository[27]. This dataset contains 800 points and has a cluster border defined by density. The experiment is done using different parameters as shown in Table 2. It is not possible to visualize all the results.

The simulation results for the map size 15×14, $\alpha(0) = 0.8$, $\alpha_a = 0.5$, $\alpha_d = 0.04$, Epoch = 20 are shown in Fig. 4. It can be seen in Fig. 4 that using E-SOM, more neurons learnt the input distribution. In E-SOM, a very less number of neurons were found to be lying in between the two diamonds. So more neurons learn the given input dataset correctly than before, thus resulting in lesser dead units. The topographic and quantization error of E-SOM is smaller than SOM and FN-SOM as shown in Table 2, which is desirable as it shows an enhancement in the learning capabilities. The neuron utilization *U* of E-SOM is larger than SOM and FN-SOM as shown in Table 2. So from the experimental results, we can conclude that E-SOM decreases dead units with fewer errors and more neuron utilization.

C. Experiment 3

We have taken the mixture of six 2-D Gaussian distributions with mean $(-2.0, -1.5)^{\mathrm{T}}$, $(-0.5, -1.0)^{\mathrm{T}}$, $(-1.5, 0.5)^{\mathrm{T}}$, $(1.0, 1.0)^{\mathrm{T}}$, $(2.0, 0.75)^{\mathrm{T}}$, and $(2.0, -1.5)^{\mathrm{T}}$ with the covariance matrices $0.02\mathbf{I}$, $0.025\mathbf{I}$, $0.005\mathbf{I}$, $0.005\mathbf{I}$, $0.02\mathbf{I}$, and $0.002\mathbf{I}$ respectively, as shown in Fig. 5., where **T** denotes the transpose operation of a matrix and **I** is the identity matrix. A dataset having 470 inputs is generated. The experiment is done using different parameters as shown in Table 3. It is not possible to visualize all the results. The simulation results for map size 13*15, $\alpha(0) = 0.9$, $\alpha_a = 0.4$, $\alpha_d = 0.04$, Epoch = 20 has been shown in Fig. 5.

The topographic and quantization error of the E-SOM are smaller than SOM and FN-SOM as shown in Table 3. The neuron utilization *U* of E-SOM is larger than the conventional SOM and FN-SOM as shown in Table 3. So from the results, it can be concluded that there is a significant improvement in the learning capabilities of neurons using E-SOM.

5. Conclusion

An efficient SOM learning algorithm to decrease dead units and improve learning efficiency of SOM has been proposed. The E-SOM makes two groups of the neurons on the map: normal and distant. The experimental results show that the neurons learnt the input data topology more precisely; and there are fewer dead units on the map using E-SOM in comparison to SOM and FN-SOM. Also, the

results have much lower quantization and topographic error on an average using E-SOM in comparison to the SOM and FN-SOM. In addition to this, E-SOM leads to higher neuron utilization on an average in comparison to the SOM and FN-SOM.

References

1. T. Kohonen, *Self-Organizing Maps*, 3rd edn. (Springer-Verlag, Berlin Heidelberg, New York, 1995).
2. Shu-Ling Shieh and I-En Liao, A new approach for data clustering and visualization using self-organizing maps, *Expert Syst. Appl.* 39 (2012) 11924–11933.
3. Y. Cheng, Clustering with Competing Self-Organizing Maps, in *Proc. Int. Joint Conf. on Neural Networks (IJCNN),* (Baltimore, Maryland, 1992), pp. 785-790.
4. W. Wan and D. Fraser, M2dSOMAP: Clustering and Classification of Remotely Sensed Imagery by Combining Multiple Kohonen Self-Organizing Maps and Associative Memory, in *Proc. of Int. Joint Conf. on Neural Networks (IJCNN),* (Piscataway, NJ, 1993), pp. 2464-2467.
5. J. Vesanto and E. Alhoniemi, Clustering of the Self-Organizing Map, *IEEE T. Neural Networ*. 11 (3) (2002) 586-600.
6. Lapidot, H. Guterman and A. Cohen, Unsupervised Speaker Recognition Based on Competition between Self-Organizing Maps, *IEEE T. Neural Networ*. 13(4) (2002) 877-887.
7. D. Desieno, Adding a conscience to competitive learning, in *Proc. of Int. Conf. on Neural Networks (ICNN),* (Piscataway, NJ, 1988), pp. 117-124.
8. L. Xu, A. Krzyzak, and E. Oja, Rival penalized competitive learning for clustering analysis, RBF net, and curve detection, *IEEE T. Neural Networ*. 4(4) (1993) 636–649.
9. 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), pp. 467–471.
10. Y. M. Cheung, On rival penalization controlled competitive learning for clustering with automatic cluster number selection, *IEEE T. Knowl. Data En.* 17(11) (2005) 1583–1588.
11. Guénaël Cabanes, Younès Bennani and Dominique Fresneau, Enriched topological learning for cluster detection and visualization, *Neural Networks* 32 (2012) 186–195.
12. N. J. Mount and 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.
13. Y. Cheung and L. Law, Rival Model Penalized Self Organizing Map, *IEEE T. Neural Networ.* 18 (1) (2007) 289-295.
14. Matsushita Haruna and Nishio Yoshifumi, Self-Organizing Map with False Neighbor Degree between Neurons for Effective Self-Organization, in *Proc. 6th Int. Workshop on Self-Organizing Maps (WSOM)*, (Bielefeld, Germany, 2007), pp. 1-6.
15. Matsushita, H. and Nishio, Y., Self-Organizing Map with Weighted Connections avoiding false-neighbor effects, in *Proc. of IEEE International Symposium on Circuits and Systems(ISCAS)*, (New York, 2010), pp. 2554-2557.
16. Riccardo Rizzo, A New Training Method for Large Self Organizing Maps, *Neural Process. Lett.* 37 (2013) 263–275.
17. Antonio Neme and Pedro Miramontes, Self-Organizing Map Formation with a Selectively Refractory Neighborhood, *Neural Process. Lett.* 39 (2013) 1-24.
18. Van Hulle M, Topology-preserving map formation achieved with a purely local unsupervised competitive learning rule, *Neural Networks* 10(3) (1997) 431–446.
19. Bamford SA and AF Murray, Synaptic rewiring for topographic map formation and receptive field development, *Neural Networks* 23 (2010) 517–527.
20. Maia JEB, Barreto GA and Coelho ALV,Visual object tracking by an evolutionary self-organizing neural network, *J. Intell. Fuzzy Syst.* 22 (2011) 69–81.
21. Iren Valovaa, George Georgievb, Natacha Gueorguievac and Jacob Olsona, Initialization Issues in Self-organizing Maps, *Procedia Computer Science* 20 (2013) 52–57.
22. H. Hasegawa, Optimization of Group Behavior, *Jpn. Ethological Soc. Newslett.* 43 (2004) 22-23.
23. Joseph P.Herbert and JingTaoYao, A Granular Computing Framework for Self-Organizing Maps, *Neurocomputing* 72 (13-15) (2009) 2865–2872.
24. T. Kohonen, *Self-Organizing Maps*, 3rd edn. (Springer-Verlag, NewYork, 2001).
25. H. Yang and M. Palaniswami, on the issue of neighborhood in self-organizing maps, in *Proc. of Symp. Appl. Computing (ACM/SIGAPP),*(Kansas USA, 1992), pp. 412-416.
26. K. Kiviluoto, Topology preservation in self-organizing map, in *Proc. of Int. Conf. on Neural Networks (ICNN),* (Washington, USA, 1996), pp. 294-299.
27. Bache, K. & Lichman, M., UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science (2013).