GASOM: Genetic Algorithm assisted Architecture Learning in Self Organizing Maps

Ashutosh Saboo, Anant Sharma, and Tirtharaj Dash

Data Science Research Group
Department of Computer Science
Birla Institute of Technology and Science Pilani
Goa Campus, Goa 403 726, India
{f2014427,f2014051,tirtharaj}@goa.bits-pilani.ac.in

Abstract. Self Organizing Map (SOM) is a special kind of neuron architecture that partially simulates the visual cortex of the animal brain and has been proven to be exceptionally successful in data visualization and clustering applications. Generally, these applications start with a predefined and fixed representation architecture of SOM without considering the underlying characteristics of the data in the original input space. In such a scenario, the performance so obtained might not be considered to be optimal. In order to enhance the quality and performance of SOM, we propose to use an evolutionary computation approach, the Genetic Algorithm (GA) to learn the optimal architecture of SOM given any data with adverse characteristics and complexity. The developed package named GASOM has been extensively evaluated with 6 synthetic datasets and 6 real-world datasets. The quality of mapping in terms of error measures have been noted carefully for each evaluation. The recorded quantitative outcomes of GASOM for each dataset demonstrate promising performance with regard to quality of mapping from the input space to the representation space.

Keywords: Data clustering, Data visualization, Evolutionary compuation, Representation space, Topographic error, Quantization error

1 Introduction

Kohonen's Self Organizing Map (SOM) [9] is one of the closest artificial neuron architecture representing in general sense, the visual cortex of the animal brain. The SOM networks are exceptionally successful in data visualization and clustering applications in which the mapping is carried out from a very high-dimensional space into either one-dimensional representation space or two-dimensional representation space. The output or representation space can also be called as the SOM space. In one-dimensional space, the neurons are arranged in the form of a one-dimensional chain in which each neuron has at most two neighbors (one to the left and another to the right). In the two-dimensional space, the neurons are arranged in a two-dimensional lattice in which the neighborhood format is commonly hexagonal or square [9,17].

One of the most inspiring benefits of SOM networks in the aforementioned application areas is that the similarity among the data as measured in the input space is preserved as fully as possible within the representation or SOM space [8]. Generally, many of the reported real-world applications of SOM start with a predefined fixed representation architecture in terms of number and arrangement of the neural processing elements prior to the training of the network [2]. In a setting in which the input data characteristics is unknown, it is quite obvious to state that it is non-trivial to determine the network architecture that would allow highly satisfying results. This problem of determining an optimal (or near-optimal) network architecture prior to the training still remains an open problem [3].

One of the solutions to the above open problem could be 'evolution' with time. In order to address this problem of determining an optimal architecture of SOM for input data with unknown characteristics, we propose to use an evolutionary computation approach. Our major contribution can be outlined as follows:

- We implement Genetic Algorithm (GA) to learn the optimal architecture of two-dimensional SOM from input data with unknown characteristics in general.
- The code has been parallelized to optimally utilize the computational resources and make the process more efficient. The overall package is named as 'GASOM'.
- The GASOM has been tested with (a) synthetically generated data, (b) available real-world data.

The rest of the paper is organized into different sections as follows: Section 2 gives background about learning in SOM, Section 3 gives a brief review of the work done in the field, Section 4 discusses the datasets and the proposed GASOM method, Section 5 presents experimental results followed by Section 6 which include concluding remarks.

2 Background

SOM is an abstract mathematical model of topographic mapping from the (visual) sensors to the cerebral cortex. During SOM learning process (mapping), the output neurons compete amongst themselves to become the winner, with the result that only one neuron is activated at any particular time instance. Such a competition can be induced by adding lateral inhibitory connections (negative feedback paths) between the neurons the SOM layer. These feedback connections force the neurons in SOM layer to organize themselves.

Self organization process in SOM involves four major components such as (1) initialization, (2) competition, (3) cooperation, and (4) adaptation. In the initialization process, the architecture of the representation map is fixed and all the connection weights are initialized with small random values. In the competition

phase, for each input pattern, the neurons compute their respective values of a 'discriminant function' which provides the basis for competition. The particular neuron with the smallest value of the discriminant function is declared the 'winner'. In the cooperation phase, the winning neuron determines the spatial location of a topological neighborhood of excited neurons, thereby providing the basis for cooperation among neighboring neurons. In the adaptation phase, the excited neurons decrease their individual values of the discriminant function in relation to the input pattern through suitable adjustment of the associated connection weights, such that the response of the winning neuron to the subsequent application of a similar input pattern is enhanced. This preserves the relative topology in the representation layer.

Learning algorithm The SOM uses a set of neurons arranged in a 2-dimensional rectangular or hexagonal lattice, to form a discrete topological mapping of an input, $\mathbf{x} \in \mathcal{R}^n$. Before the learning starts, all the weights $\mathbf{w} = \{w_1, w_2, \cdots, w_m\}$ are initialized to small random numbers. The weight w_i is the weight vector associated with the pattern i and is a vector of same dimension (n) as input; m is the total number of neurons, Ω is the set of indices of the neurons in the map and $h(v, k, t)$ is the neighborhood function. Algorithm 1 presents the algorithm.

Algorithm 1 SOM learning algorithm

```
1: procedure SOM–Learning
2:     while map is not converged do
3:         At each time t, present an input x(t), and select the winner
4:         Update the weights of the winner and its neighbors
5:     end while
6: end procedure
```

A winner neuron in the map is chosen using the following equation.

$$v(t) = \arg\min_{k \in \Omega} \parallel x(t) - w_k(t) \parallel \tag{1}$$

The term $\parallel x(t) - w_k(t) \parallel$ in Equation 1 is the Euclidean distance between $x(t)$ and $w_k(t)$ given in Equation 2. It should be noted that both $\mathbf{x}$ and $\mathbf{w}$ have dimension n.

$$\parallel \mathbf{x}(t) - \mathbf{w}_k(t) \parallel = \left[\sum_i \left(x_i(t) - w_{ki}(t) \right)^2 \right]^{\frac{1}{2}} \tag{2}$$

The weights related to the neighbors of a winning neuron v are updated by using Equation 3.

$$\Delta \mathbf{w}_k(t) = \alpha(t) h(v, k, t) \left[\mathbf{x}(t) - \mathbf{w}_v(t) \right] \tag{3}$$

The coefficient $\alpha(t)$, termed the 'adaptation gain' or 'learning rate', are scalar-valued, decrease monotonically with time, t; $t \geq 0$. The typical value of $\alpha(t)$ falls in the range (0,1). The neighborhood function $h(\cdot)$ is a Gaussian function.

3 Related Works

Some studies have attempted to optimize the SOM using Genetic Algorithm (GA) [5] to learn optimal parameters such as learning rate for some specific problems [4]. GA has also been applied to solve traveling salesman (TSP) problem using SOM [7]. In this study, the winning neuron is first dragged to the winning city and then pushed to the convex hull of the TSP and moved further to find out the optimal path that visits each city exactly once. The SOM has been tuned to learn faster using k-means in the initial step to find out the number of neurons to be arranged in the lattice [14]. Su et al. [15] discuss an efficient initialization of SOM representation lattice. In their algorithm, the neurons at the four corners of the lattice are initialized first and then the other neurons of the lattice. A detailed discussion on various issues during initialization phase of the SOM has been discussed by Valova et al. [16]. We believe that our present on the learning of the optimal architecture of SOM using GA is first of its kind and would aid more efficient real-world applications of SOM in robotics [6] and bioinformatics [13].

4 Materials and Methods

4.1 Dataset

For our model implementation, we have experimented with (a) six synthesized data in two-dimension and (b) six real-world data obtained from UCI machine learning (ML) repository [1]. It should be noted that the SOM learning is unsupervised and there is no relation between class labels available in the datasets with the learning process. Hence, the class labels have been ignored for this work. Table 1 presents a summary of the synthesized datasets and the real-world datasets with various dimensions obtained from UCI ML repository. Although the synthesized data are two-dimensional, these are very complex with their distribution (see Figure 1). The purpose is to test the implemented GASOM with adverse datasets.

Table 1. Dataset summary (a) Synthetic datasets, (b) Real-world datasets

Dataset	#patterns	#dimension
Corner	1000	2
crescentFullmoon	1000	2
gingerBreadman	2000	2
HalfKernel	1000	2
Outliers	600	2
TwoSpirals	2000	2

Dataset	#patterns	#dimension
Abalone	4177	8
CarEvaluation	1728	6
Glass	214	10
IrisFlower	150	4
Sonar	208	60
Wine	178	13

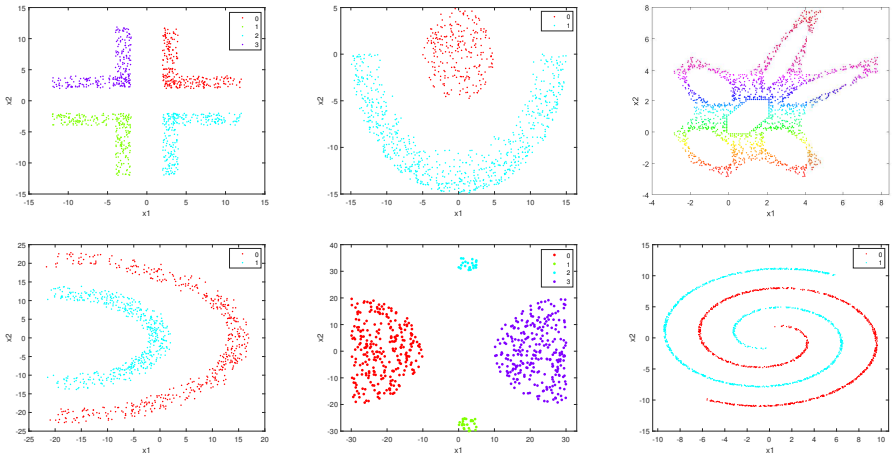

Fig. 1. Scatter plot for synthesized data (Top-left: Corner, Top-middle: CrescentFullmoon, Top-right: gingerBreadman, Bottom-left: HalfKernel, Bottom-middle: Outliers, Bottom-right: TwoSpirals)

4.2 GASOM

In our present work, the motivation behind the concept of natural evolution comes from the way search happens in other machine learning models [10] with one major difference. In conventional search over a lattice of models, the search starts from the first layer and move down towards the more specialized models and these specialized models are essentially the co-products formed by the addition of further knowledge to the generalized models. But, in our proposed GA-based search, the each layer of the lattice is a combination of generalized and specialized models of the SOM.

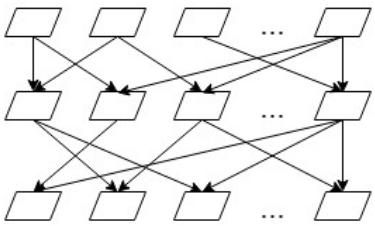

Fig. 2. Search in GASOM. Each node shown as a parallelogram is a SOM. Arrows show crossover between two SOMs to form a new SOM

The architecture of a SOM is defined as the size of the two-dimensional grid in which the neurons are arranged. The SOM models in layer i are formed by the crossover between different SOM models present at the layer $i-1$. The

search for an optimal model stops once a best-so-far model is obtained at some layer $i + k$. A relevant diagram demonstrating the process of search has been shown in the Figure 2. Each layer represents a set of candidate solutions for the quested optimal SOM. The candidate solutions from the layer i essentially propagate their properties to the next layer to generate evolved SOM models which are better in representing the characteristics of the data available in the input space. During the process of evolution, a set of SOM models at each layer would be discarded which are unfit as compared to other SOM models. The fitness for each SOM can be computed from their qualitative and quantitative representation of the input space in the representation space. To measure the quality (fitness) of the mapping, we used the average of the quantization error (QE) and topographic error (TE) which are explained in the following equations.

The quantization error (E_{QE}) is a measure of the quality of adaptiveness of the SOM. It is essentially the average distance between input data vector $\mathbf{x}_p$ and its winner $\mathbf{v}_p$ where $1 \leq p \leq m$; m is the total number of data patterns in the input data space. The topographic error (E_{TE}) shows how well the trained SOM network preserves the topography of the data mapped onto it. Equation 4 and Equation 5 present the quantization error and the topographic error respectively. We define the fitness of a SOM as the average of these two error as given in the Equation 6.

$$E_{QE} = \frac{1}{m} \sum_{p=1}^{m} \parallel \mathbf{x}_p - \mathbf{v}_p \parallel \tag{4}$$

$$E_{TE} = \frac{1}{m} \sum_{p=1}^{m} \varphi\left(\mathbf{x}_p\right) \tag{5}$$

$\varphi(\mathbf{x}_p)$: For a data point $\mathbf{x}_p$, if two best matching units (the winner neuron and the runners-up neuron) are adjacent in the map, then $\varphi(\mathbf{x}_p) = 0$; otherwise $\varphi(\mathbf{x}_p) = 1$.

$$fitness = \frac{E_{QE} + E_{TE}}{2} \tag{6}$$

The flowchart of the major processes of the GASOM is shown in the Figure 3. The input data is loaded at first. The GA parameters such as population size, the initial search space, number of evolution(generation), crossover probability, elitism are initialized in the GA module. For each generation, the SOM is run once for each candidate solution to test its quality on the input data. The process is continued until the optimal SOM is obtained. The independent subprocesses such as SOM evaluation for each candidate solution in a generation has been parallelized to reduce the expenditure of computational resources. The SOM algorithm follows as given in the Algorithm 1.

5 Experimental Results

Environment All the simulations have been conducted in High Performance Computing Cluster system with 14 processors and each machine with 32GB

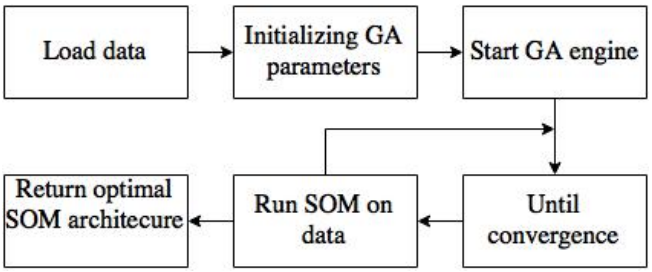

Fig. 3. Flowchart of the major processes involved in GASOM

of main memory. The full package of the codes have been implemented using Python which uses the standard libraries such as numpy, pandas and necessary modification to Pyevolve [12] and SOMPY [11] to support this research. The code will be made publicly available in the link `URLofGASOM`.

Parameters The parameters in GA are described as follows: The population size is set to 10, crossover is single point with rate 0.80, mutation is swap-based mutation with rate 0.02, elitism is set to TRUE with value 2, the number of maximum evolution (generation) in GA is set to 15. The search space for GA has been defined based on a density measure. The density is defined as $\frac{m}{lx \times ly}$, where lx and ly are the number of neurons in two sides of the rectangular lattice of the SOM map. We test the model for three different search spaces satisfying (a) density=0.5, (b) density=2, and (c) no restriction on density i.e default search space is $[1, 100]$ for both lx and ly. The SOM parameters are as follows: The training is batch-training that includes rough training and fine-tuning, the maximum iteration is set to 1000 (800+200), initial weight configuration is randomized, the neighborhood function $h(\cdot)$ is Gaussian. After obtaining the optimal architecture from the machine, the SOM is once again run for 2000 (1600+400) iterations to record the final quality of the mapping.

Results The results recorded by implementing GASOM for both synthetic data and the real-world data have been reported in the Table 2 and Table 3 respectively. The table depicts the optimal two-dimensional SOM architecture and the corresponding errors suggesting the quality of mapping and the quality of topology preservation. The results provide a lot of insights into these two important factor and the optimal architecture of the SOM. It can be seen that by restricting initially the density to a fixed value improves the quality of mapping by restricting the search space for GA thereby obtaining the best quality SOM with near-optimal architecture.

Further to the above inferences, it could also be observed that overall performance of GASOM for the real-world dataset is slightly lower than that for the synthesized datasets with regard to the obtained fitness. One possible and obvious reason behind such performance could be the dimension of the synthetic data. Since the synthesized dataset is two-dimensional, the relative closeness of the SOM space and the input space is higher than the closeness achieved for

higher dimensional data in real-world datasets. However, GASOM needs further improvements to work efficiently for the high dimensional data such as Sonar which has 60 features.

Table 2. Performance of optimal SOM for synthetic data

Dataset	Initial density	Optimal architecture $[lx, ly]$	E_{QE}	E_{TE}	$fitness$	Final density
Corner	0.5	[35, 45]	0.0193	0.0110	0.0151	0.6349
	2.0	[19, 23]	0.0470	0.0060	0.0265	2.28
	none	[20, 16]	0.0574	0.0110	0.0342	3.12
crescentFullmoon	0.5	[45, 11]	0.0474	0.0220	0.0347	2.02
	2.0	[12, 23]	0.0618	0.0310	0.0464	3.62
	none	[14, 20]	0.0625	0.0250	0.0437	3.57
gingerBreadman	0.5	[63, 26]	0.0244	0.0680	0.0462	1.22
	2.0	[22, 21]	0.0519	0.0940	0.0729	4.32
	none	[19, 19]	0.0606	0.0745	0.0675	5.54
HalfKernel	0.5	[38, 32]	0.0249	0.0350	0.0299	0.82
	2.0	[22, 22]	0.0454	0.0230	0.0342	2.06
	none	[16, 18]	0.0639	0.0150	0.0394	3.47
Outliers	0.5	[30, 35]	0.0248	0.0500	0.0374	0.5714
	2.0	[15, 15]	0.0713	0.0650	0.0681	2.66
	none	[19, 17]	0.0598	0.0750	0.0674	1.85
TwoSpirals	0.5	[36, 43]	0.0119	0.0590	0.0354	1.29
	2.0	[17, 27	0.0324	0.0025	0.0174	4.35
	none	[19, 14]	0.0567	0.0070	0.0318	7.51

6 Conclusion

Here we investigate the application of an evolutionary heuristic, the Genetic Algorithm (GA) to search for an optimal architecture of SOM given any input data with complex characteristics is presented. The resulting model called GASOM has been extensively tested for 6 synthetic datasets and 6 real-world datasets. The quality of mapping in terms of error measure has been considered for different density parameters coined in this work. The results obtained by GASOM demonstrate qualitative performance with regard to learning and mapping the data from input space to the representation space of SOM. This study would aid in many different real-life applications such as robotic arm control and bioinformatics.

Table 3. Performance of optimal SOM for real-world data

Dataset	Initial density	Optimal architecture $[lx, ly]$	E_{QE}	E_{TE}	$fitness$	Final density
Abalone	0.5	[26, 84]	0.2950	0.1541	0.2246	1.91
	2.0	[23, 24]	0.3939	0.0802	0.2370	7.56
	none	[19, 69]	0.3328	0.0100	0.1714	3.18
CarEvaluation	0.5	[47, 47]	0.9237	0.2152	0.5694	0.78
	2.0	[13, 19]	1.1548	0.2384	0.6966	6.99
	none	[64, 64]	0.8972	0.0833	0.4902	0.42
Glass	0.5	[20, 13]	0.6658	0.0000	0.3329	0.82
	2.0	[11, 11]	0.9399	0.0000	0.4699	1.76
	none	[98, 98]	8.55e-7	1.000	0.5000	0.02
IrisFlower	0.5	[18, 18]	0.1719	0.0000	0.0859	0.46
	2.0	[9, 9]	0.3577	0.0000	0.1788	1.85
	none	[43, 100]	1.3428	0.0866	0.0433	0.03
Sonar	0.5	[18, 20]	3.4226	0.4519	1.9373	0.57
	2.0	[8, 10]	4.8568	0.4326	2.6447	2.60
	none	[52, 100]	0.0284	0.9134	0.4709	0.04
Wine	0.5	[16, 16]	1.3223	0.0000	0.6611	0.69
	2.0	[10, 8]	1.7823	0.0000	0.8911	2.22
	none	[79, 98]	1.12e-6	0.0000	5.62e-7	0.02

References

1. Asuncion, A., Newman, D.: Uci machine learning repository (2007)
2. Dittenbach, M., Merkl, D., Rauber, A.: The growing hierarchical self-organizing map. In: Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on. vol. 6, pp. 15–19. IEEE (2000)
3. Dittenbach, M., Rauber, A., Merkl, D.: Recent advances with the growing hierarchical self-organizing map. In: Advances in Self-Organising Maps, pp. 140–145. Springer (2001)
4. Harp, S.A., Samad, T.: Genetic optimization of self-organizing feature maps. In: Neural Networks, 1991., IJCNN-91-Seattle International Joint Conference on. vol. 1, pp. 341–346. IEEE (1991)
5. Holland, J.H.: Genetic algorithms. Scientific american 267(1), 66–72 (1992)
6. Huang, D.W., Gentili, R.J., Katz, G.E., Reggia, J.A.: A limit-cycle self-organizing map architecture for stable arm control. Neural Networks 85, 165–181 (2017)
7. Jin, H.D., Leung, K.S., Wong, M.L., Xu, Z.B.: An efficient self-organizing map designed by genetic algorithms for the traveling salesman problem. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 33(6), 877–888 (2003)
8. Kiviluoto, K.: Topology preservation in self-organizing maps. In: Neural Networks, 1996., IEEE International Conference on. vol. 1, pp. 294–299. IEEE (1996)

9. Kohonen, T.: The self-organizing map. Neurocomputing 21(1), 1–6 (1998)
10. Mitchell, T.M., et al.: Machine learning (1997)
11. Moosavi, V.: Contextual mapping: Visualization of high-dimensional spatial patterns in a single geo-map. Computers, Environment and Urban Systems 61, 1–12 (2017)
12. Perone, C.S.: Pyevolve: a python open-source framework for genetic algorithms. ACM SIGEVOlution 4(1), 12–20 (2009)
13. Polat, O., Dokur, Z.: Protein fold recognition using self-organizing map neural network. Current Bioinformatics 11(4), 451–458 (2016)
14. Su, M.C., Chang, H.T.: Fast self-organizing feature map algorithm. IEEE Transactions on Neural Networks 11(3), 721–733 (2000)
15. Su, M.C., Liu, T.K., Chang, H.T.: An efficient initialization scheme for the self-organizing feature map algorithm. In: Neural Networks, 1999. IJCNN'99. International Joint Conference on. vol. 3, pp. 1906–1910. IEEE (1999)
16. Valova, I., Georgiev, G., Gueorguieva, N., Olson, J.: Initialization issues in self-organizing maps. Procedia Computer Science 20, 52–57 (2013)
17. Vesanto, J., Alhoniemi, E.: Clustering of the self-organizing map. IEEE Transactions on neural networks 11(3), 586–600 (2000)