American Journal of Information Science and Computer Engineering
Vol. 3, No. 6, 2017, pp. 77-83
http://www.aiscience.org/journal/ajisce
ISSN: 2381-7488 (Print); ISSN: 2381-7496 (Online)

Optimisation of Self Organising Maps Using the Bat Algorithm

Kernan Mzelikahle[1, *], Dunstan Junior Mapuma[1], Dumisani John Hlatywayo[2], John Trimble[3]

[1]Computer Science Department, National University of Science and Technology, Bulawayo, Zimbabwe
[2]Applied Physics Department, National University of Science and Technology, Bulawayo, Zimbabwe
[3]Industrial Engineering Department, Tshwane University of Technology, Tshwane, South Africa

Abstract

Self Organising Maps are among the most widely used unsupervised neural network approaches to clustering. They have been shown to be efficient in handling large and high dimensional data. The Bat Algorithm is a swarm intelligence based, meta-heuristic optimisation algorithm. It is based on the echolocation behaviour of micro-bats with varying emission pulse rates and loudness. This paper gives a novel hybrid optimisation method which is here called the Bat Optimised Self-Organising Map. It combines the basic Self Organising Map learning algorithm with the Bat Algorithm. In this optimisation technique, the Bat Algorithm is used to initialise the weight vectors for a Self Organising Map to a near global optimum prior to the competition.

Keywords

Self Organising Maps, Bat Algorithm, Artificial Neural Networks, Unsupervised Learning

Received: October 23, 2017 / Accepted: November 9, 2017 / Published online: December 9, 2017

1. Introduction

A Self Organising Map (SOM) is a type of an Artificial Neural Network (ANN) that is trained using unsupervised learning methods [5]. An ANN is a collection of highly interconnected neurons that transform a set of inputs to a desired set of outputs. It is inspired by the mammalian biological nervous system [11]. Fundamental concepts of SOMs were first introduced by Malsburg [12] and later extended by Kohonen [10] as an attempt to mimic the apparent actions of a small class of biological neural networks that are responsible for sensory processing. The idea was to create an ANN that could learn without supervision, an abstract representation of some sensory input. In unsupervised ANN learning, the connection weight adjustments are not made by comparison to some target output as there is no teaching signal to control the weight adjustments. The network, instead, is trained by being shown the patterns that are to be classified, and it is allowed to produce its own output representation for the classification. The network must discover patterns, regularities, features, clusters and adjust its parameters simultaneously. This property is known as 'self-organisation'.

SOMs are different from regular ANNs in that they apply undirected learning methods as opposed to directed learning methods such as the back-propagation algorithm, and the reinforcement learning method [14]. The most common learning method used in SOMs is the competitive learning algorithm. Generally, output neurons are in competition with one another over input patterns [4]. The competitive learning algorithm is motivated by the anatomical and physiological evidence of lateral interaction between neurons in mammalian nervous systems [10]. In the competitive learning algorithm, those neurons which respond strongly to

* Corresponding author
E-mail address: kernan.mzelikahle@nust.ac.zw (K. Mzelikahle)

input stimuli have their weights updated. When an input pattern is presented to a SOM running under competitive learning, all neurons in the input layer compete and the winning neuron undergoes weights adjustment for all its connections. The decision of whether only one neuron shall have its connection weights adjusted (winner takes all), or whether a group of neighbouring neurons shall also have their connection weights adjusted, lies with a particular model of the competitive learning process. SOMs work by increasing the specialisation of each neuron in the network, thus they are well suited to finding clusters within data [13].

One of the limitations of SOMs, as noted by Ajith et al. [1], is that SOM training often results in topologically twisted maps. The learning process sometimes takes multi-stable states within which the map is trapped to an undesirable disordered state, resulting in topological defects on the map [2]. Multi-stability is a serious problem in the learning process [6]. When the learning process is trapped in these states, the map seems to have practically converged to the final state. The existence of the topological defect critically aggravates the performance of SOMs. Once a topological defect appears in a map during the learning process, it tends to remain there for a long time and slows down the ordering process of a SOM [9]. The presence of a topological defect in a SOM has an effect of making the map settle on a local optimum, hence limiting its practical applicability. One of the main reasons why SOM learning tends to get stuck at a local optimum, is because of the initial weight vector. Generally, the initial weight vector is comprised of random weights. In this work, we propose a novel hybrid optimisation technique by combining the basic competitive learning algorithm with the Bat Algorithm. This hybrid optimisation technique is used to initialise the weight vectors for a SOM. The objective is to push the SOM to begin its competition near a global optimum.

2. The Basic SOM Algorithm

A basic SOM consists of two layers, the input layer (a holding point for the input data), and the mapping layer as shown in Figure 1. The input layer has as many neurons as there are input variables. The two layers are fully connected to each other and each of the neurons in the mapping layer has an associated weight vector $\vec{w}_i$, with one weight for each connection with the input layer. The aim of the SOM is to group like input vectors, $\vec{x}_i$, together on the mapping layer, y_j [3]. Therefore the method is topology preserving as items which are close in the input space are also close in the mapping space. During training the input vectors, $\vec{x}_i$, are presented to the SOM through the input layer in time steps, t.

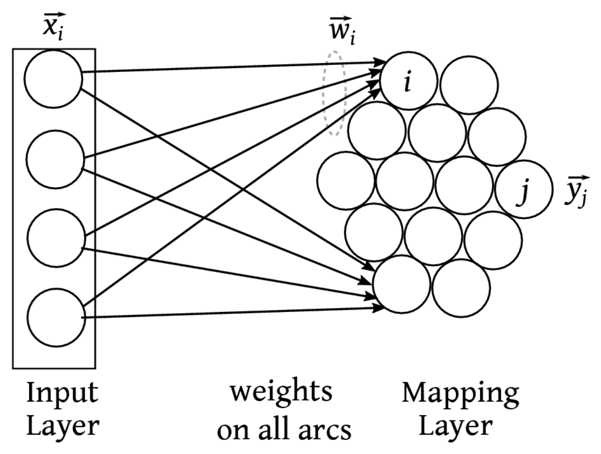


Figure 1. A sample SOM with a 2-D mapping layer (adapted from Gurney [8]).

The neurons in the mapping layer compete for the input vector. The winner is the mapping neuron whose vector of incoming connection weights most closely resembles the components of the input vector. The winner has its weight vector adjusted to relatively move towards the input vector. In hard competition (winner-takes-all), only the winner's weight vector is updated. In soft competition, the winning neuron and its neighbours within the cut-off radius of excitation, r_*, are updated in relation to their position, with the winning neuron receiving the most update. In general, the radius r is calculated based on the Euclidean distance. For the neurons within the cut-off radius, $r \le r_*$, the learning signal is excitatory, connections are strengthened. For the neurons beyond the cut-off radius, $r > r_*$, the learning signal becomes inhibitory, connections are weakened. As more input vectors are passed through the network, the weight vectors of the mapping layer neurons self-organise. By the end of the training process, different parts of the mapping layer respond strongly to specific regions of the input space (clusters). Once the training of the network is complete, the clusters obtained can be examined in order to gain better insight into the underlying data-set.

3. The Basic Bat Algorithm

The Bat Algorithm (BA) is a swarm intelligence algorithm proposed by Xin-She Yang [15], inspired by the echolocation phenomenon in bats. Based on the behaviour of bats, the Bat Algorithm updates the velocity and position iteratively. In Yang's derivation [15], the frequency f_i, of every bat, with velocity v_i, at location x_i, and wavelength λ, of signals are given by:

$$\lambda = v / f, \tag{1}$$

$$f_i = f_{min} + \left(f_{max} - f_{min}\right)\beta, \tag{2}$$

$$v_i^t = v_i^{t-1} + \left(x_i^{t-1} - x_*\right) f_i, \qquad (3)$$

$$x_i^t = x_i^{t-1} + v_i^t, \qquad (4)$$

where λ, is the wavelength of the emitted pulse signals, f_{max}, is the maximum possible frequency of pulse emissions, f_{min}, is the minimum possible frequency of pulse emissions, x_*, is the current global best location (or solution), and $\beta \in [0,1]$ is a random vector drawn from a uniform distribution. The Bat Algorithm and its pseudo-code are presented by Yang and Gandomi [16]. In their work, they presented a reduction of the bat biological nature and its hunting processes into a computer implementable algorithm. They make a number of assumptions in this reduction which render it limited, however, the algorithm has been shown to be effective for computational purposes.

4. The Proposed SOM Optimisation Method

The proposed SOM Optimisation method is a fusion of the bat algorithm and SOM, herein referred to as Bat Optimised Self-Organising Map (BOSOM). In this method there are three stages:

i. all connections to neurons in a SOM are initialised to some random weight w_{ij}, such that, $w_{ij} \in \Re; w_{ij} = (0,1)$

ii. the bat algorithm is used to adjust the weights seeking a global optimum set of weights using equations (2) to (4), and

iii. the competitive learning method is then used to train the network starting off with this set of bat algorithm optimised weights.

For the Bat Algorithm to work successfully, it needs an objective function that defines an acceptable solution. In this study, the following was used:

$$f\left(\vec{w}_j(t+1)\right) = \frac{\vec{w}_j(t) + \gamma\left(\vec{x}(t) - \vec{w}_j(t)\right) - w_j^2 y_j}{\| \vec{w}_j(t) + \gamma\left(\vec{x}(t) - \vec{w}_j(t)\right) - w_j^2 y_j \|}, \qquad (5)$$

where γ is the learning rate, $\vec{w}_j(t)$ is the weight vector on neuron j at some time t, and y_j is the output of neuron j calculated as $y_j = \sum_i w_{ij} x_i$. The objective function $f(\vec{w}_j)$ is normalised such that the weight vector can be controlled. In the experiments, the input data did not need to be normalised in turn because real-life data may not always be normalisable before use. An acceptable solution is determined at either 1%, or 5%, or 10% confidence intervals; that is, $(f(\vec{w}_j(t+1)) - f(\vec{w}_j(t))) \le 0.01$ or $(f(\vec{w}_j(t+1)) - f(\vec{w}_j(t))) \le 0.05$ or $(f(\vec{w}_j(t+1)) - f(\vec{w}_j(t))) \le 0.10$, respectively. The pseudo-code for the proposed optimisation method, BOSOM, is given in Algorithm 1. In Algorithm 1, sub-numbering for the standard SOM is not used because the standard SOM is found in literature.

Table 1. Listing of BOSOM Algorithm.

```
Algorithm 1: Pseudo-code for our proposed BOSOM Method
BEGIN
1.  Initialise $\vec{w}_j$ randomly for all $j$, where $j$ is a neuron; such that
    $w_j \in \Re; w_j = (0,1)$
2.  Set BAT parameters, and topological parameters
3.  Define the objective function $f(w_{ij})$ and set acceptable confidence
    threshold e.g. $\varphi = 0.05$
4.  FOR (EPOCHS < MAX)
    WHILE ( $\left|f\left(w_{ij}(t+1)\right) - f\left(w_{ij}(t)\right)\right| > \varphi$ ) DO
        4.1 Generate new solutions by adjusting frequency, updating
            velocities, and locations of virtual bats using equations (2)
            to (4).
        4.2 Generate new weights using; $w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}$,
            where: $\Delta w_{ij} = \gamma x_i\left(y_j - y_j^2 w_{ij}\right)$,
            and: $y = \sum_i w_{ij} x_i$
    END WHILE
    END FOR
5.  RUN STANDARD SOM COMPETITION:
    WHILE(< EPOCHS | ERROR CONVERGED >) DO
    FOR (Each input vector, $\vec{x}$ )
    FOR (each neuron $j$, compute )
    $D(j) = \sum_i (w_{ij} - x)^2$
    END FOR
    Find neuron $j_*$ such that $D(j_*)$ is minimum
    FOR (all neurons $j$ in specified neighbourhood of $j_*$ and
    for all $i$ )
    Compute new competitive weights: $w_{ij}(t+1) = w_{ij}(t) + \gamma\left(x_i - w_{ij}(t)\right)$
    END FOR
    Update learning rate, $\gamma$;
    Reduce radius $r$ of topological neighbourhood at specified times
    END FOR
    END WHILE
END
```

5. Experiments and Results

In order to assess the utility of the BOSOM method, one clustering problem from the University of California, Irvine

(UCI) Machine Learning Repository, was examined. Experiments were conducted using the Iris data-set. Comparisons were then made between the standard SOM and BOSOM. Both the standard SOM and the BOSOM were trained using a standard square lattice structure for a self organising network. Figure 2 shows a generalised structure of the network used in this study.

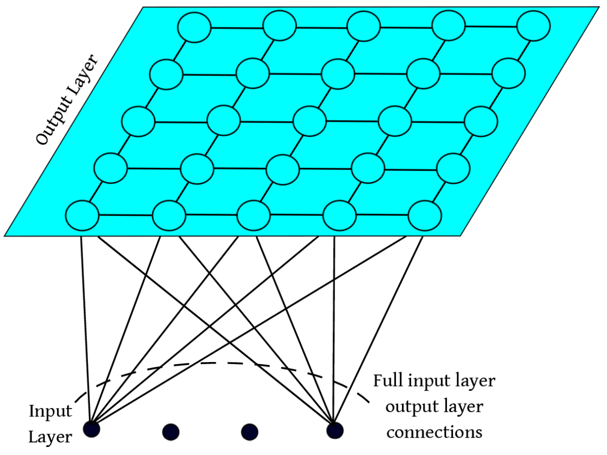


Figure 2. SOM Lattice structure used in this study.

5.1. The Iris Data-Set

The data-set we used to train and test SOM and BOSOM performance for clustering is of known parameters and expected outcomes. The Iris data-set is a small, well-understood and known data-set that consists of the measurements of four attributes of 150 Iris flowers from three iris species. The four data dimensions are: the sepal length in centimetres, the sepal width in centimetres, the petal length in centimetres and the petal width in centimetres. The three iris species are Iris Setosa, Iris Versicolour and Iris Virginica. Each type of Iris has 50 samples as well as some properties about each flower. One flower species is linearly separable from the other two, yet these other two are not linearly separable from each other.

Table 2. Data and Class Information.

Data type	Iris
Input node	4
Data Size	150
Training Size	120
Testing Size	30
Number of Classes	3

The typical tendency of researchers for the Iris data-set is to cluster Iris types based on the measurements. It is one of the most analysed data-sets in statistics, data mining, and multivariate visualisation. It was first published by Fisher [7], and is widely available. Table 2 shows a summary of the data-set class information.

5.2. Performance Evaluation

The goal of the conducted experiments was to investigate the performance of the proposed method. Comparisons were done between standard SOM and BOSOM. The results were validated in terms of Clustering Accuracy (CA) and Quantisation Error (QE) on the Iris data-set used universally as a machine learning data-set. QE is measured after a network's training and CA is the analysis for testing. The efficiency of the proposed method is validated accordingly; that is, if QE values are small and CA values high, then the results are promising to be good. QE is used for measuring the quality of the network. QE of an input vector is defined by the difference between the input vector and the closest codebook vectors. QE describes how accurately the neurons respond to the given data-set. Thus the quantisation error is used for the computation of similarity of patterns. The QE calculation in this study was carried out as;

$$E_q = \frac{1}{N}\sum_{k-1}^{N}\left(x_k(t) - w_{mk}(t)\right) \tag{6}$$

where w_{mk} is the best unit of weight at time t. On the other hand, the Clustering Accuracy (CA) indicates how well the classes are separated on the map. The CA of new samples measures the network's ability to generalise. In this study, the clustering accuracy calculation, as a percentage, is given by;

$$P = \frac{n}{N} \times 100, \tag{7}$$

where n is the number of correctly clustered patterns and N is the total number of patterns in the testing data.

5.3. Results and Discussion

The network architecture used for both the BOSOM and standard SOM consist of 4 input nodes and a 5×5 output lattice grid. One hundred and twenty data patterns were used to train the network. For bat optimisation; the parameters are, loudness A = 0.5, the pulse rate $r_0 = 0.5$, minimum frequency $f_{min} = 0.0$, and maximum frequency $f_{max} = 0.1$. The bat population was set at 500 and problem dimension as 25 (a 5×5 grid-structure) with maximum epochs of 2000. The input vectors $\vec{x}_i$ were serially presented to the input layer. The number of input nodes corresponded to the number of data units in the input vectors that were fed into the network, while the number of nodes in the output layer represented the maximum number of possible classes. Table 3 shows the parameters that were used for both the SOM and BOSOM.

Table 3. Parameter Settings for both SOM and BOSOM.

Parameter	Iris
Input vectors (Training)	120
Input vectors (Testing)	30
Input dimension	4
Mapping Dimensions (X,Y)	5×5
Lattice structure	Square Topology
Learning rate	0.5
Pulse rate (BOSOM only)	0.5
Number of runs	10 times
Epochs	2000
Number of bats (BOSOM only)	500
Minimum frequency	0.0
Maximum frequency	0.1
Stop condition (minimum error)	0.0000193

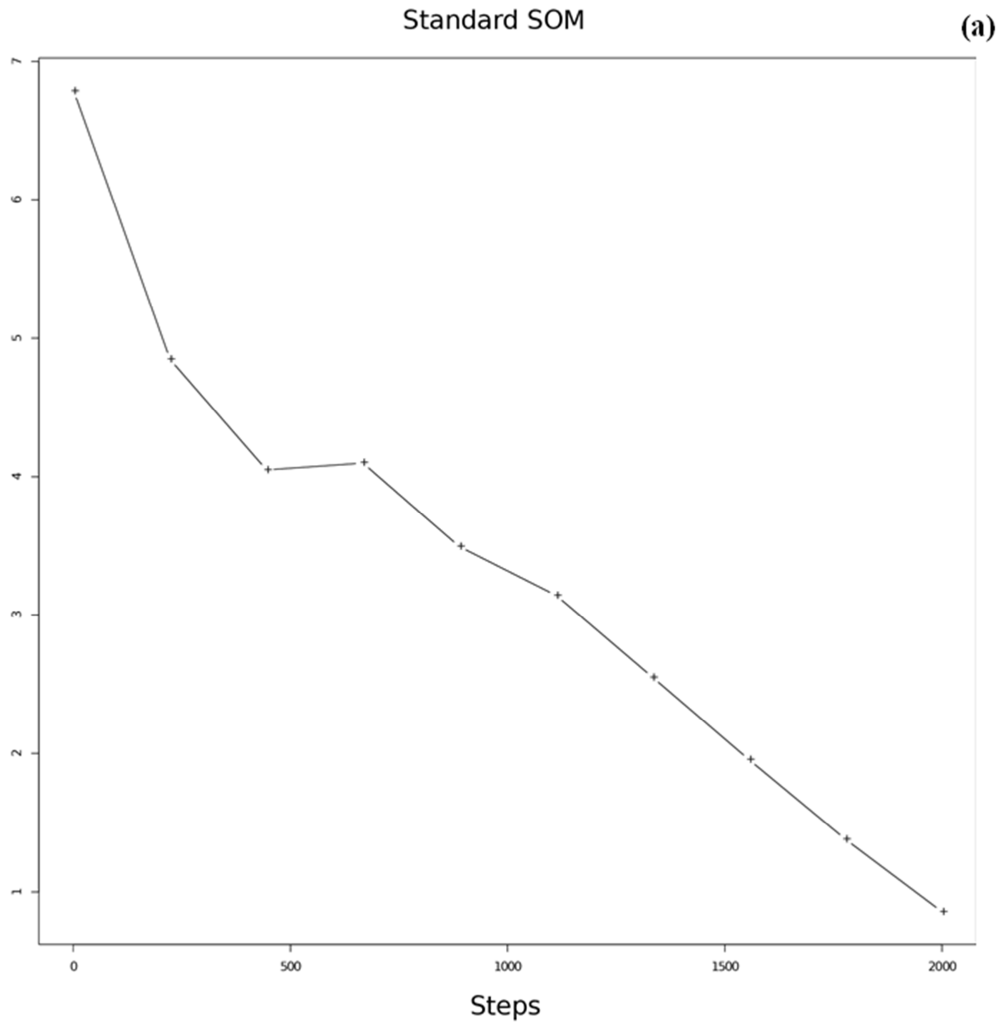
The acceptable stopping condition, in the form of acceptable error, was arrived at from empirical trials. The topographic error value varies between 0 (good projection quality) and 1 (poor projection quality). The topographic quality of the mapping in this study was equal to 0.05; which meant that all observations had a second best unit which was in the neighbourhood of the best matching unit.

The quantisation error is an unbounded positive number. The closer it is to 0, the better the projection quality. As can be drawn from the results in Table 4, the quantisation error for BOSOM was 0.0171 and was closer to 0 than for SOM at 0.0348. Also BOSOM had a 76.6667 clustering accuracy, statistically better than SOM with 74.3333 clustering accuracy. This led to the conclusion that BOSOM out performed SOM in the experiment.

Table 4. Summary of SOM and BOSOM results.

	SOM	BOSOM
Epochs	2000	2000
Convergence Error	0.0318	0.0243
Quantisation Error	0.0348	0.0171
Classification (%)	74.3333	76.6667

From Table 4, correct classification percentage shows that BOSOM results were better than SOM by a 2.334% margin. This margin is statistically significant in ANN training. For Iris learning, both algorithms converge using the maximum number of pre-specified iterations.



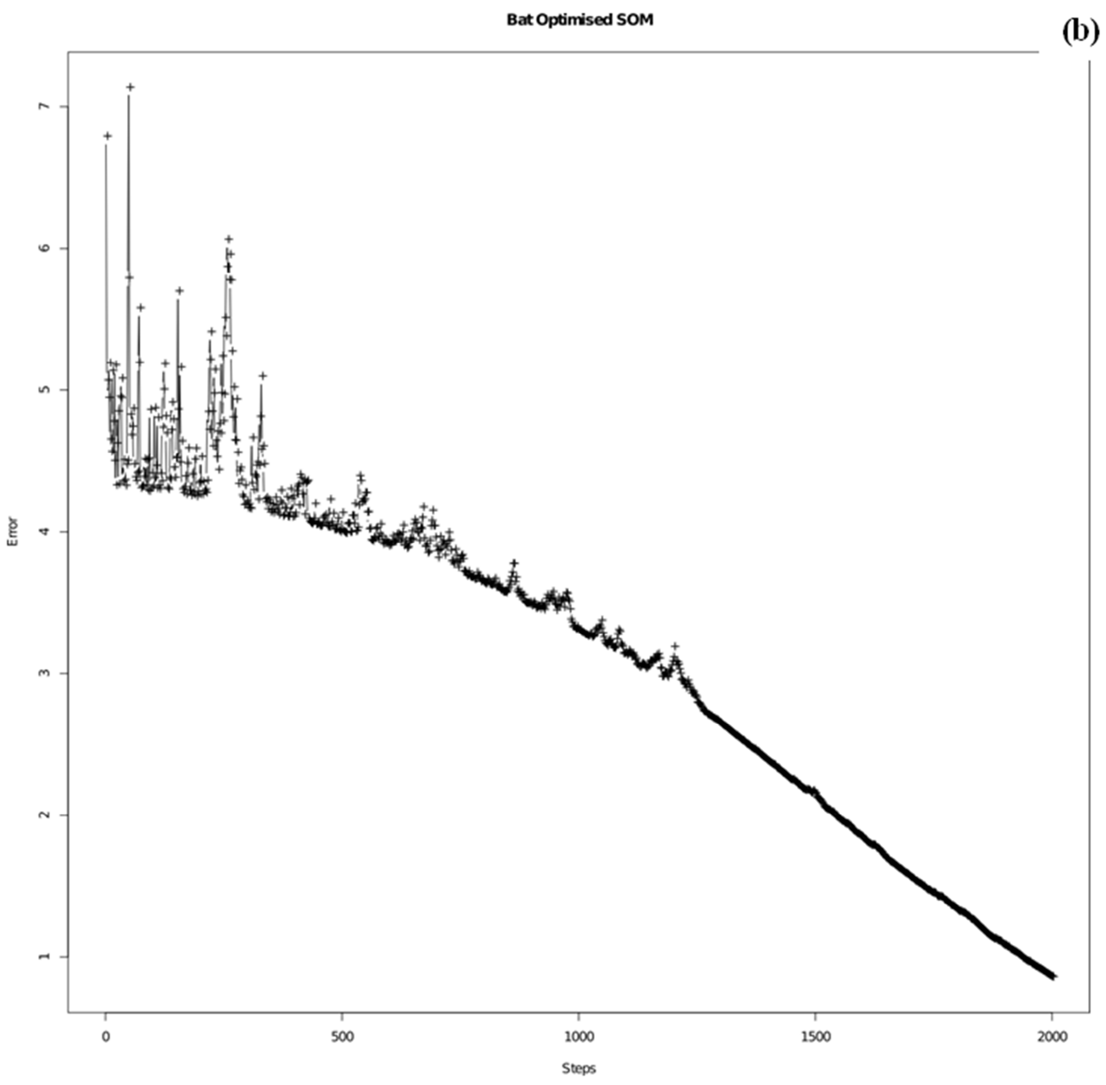


Figure 3. Convergence of SOM compared to BOSOM.

For 2000 iterations, SOM converges at a minimum error of 0.0318 while BOSOM converges at a minimum error of 0.0243. Figure 3 shows a comparison of error convergence between the SOM and BOSOM under same empirical circumstances. Figure 3 (a) shows the behaviour of error during training until a stopping condition, or maximum number of epochs, is met. The results show that the quantisation error steadily declines with minimal sensitivity as the SOM approaches the stopping condition or maximum number of epochs at 2000 epochs. The standard SOM does not show sufficient sensitivity to input, and it is not clear why this is so. Figure 3 (b) shows the quantisation error for BOSOM. It clearly shows good sensitivity to new input and good error decline. It may be put forth that, in BOSOM the virtual bats work together to find the lowest error at each iteration and therefor consistently reduce the error at each iteration.

6. Conclusion

By only adjusting the initial weights of the standard SOM, the BOSOM shows interesting results that need to be investigated further. It appears that the performance of the BOSOM is statistically better than the standard SOM. This may be attributed to the fact that BOSOM initialises the initial weight vector near a global optimum. Further research needs to be done, to establish whether the BOSOM could be further improved by regulating the learning rate, γ . This regulation may contribute to BOSOM being smoother and faster.

References

[1] Ajith, A., Aboul-Ella, H., and Carvalho, A. (2009). Foundations of Computational Intelligence Volume 4: Bio-Inspired Data Mining: Studies in Computational Intelligence, Germany, Springer Verlag.

[2] Aoki, T., Ota, K., Kurata, K. and Aoyagi, T. (2009). Ordering Process of Self-Organizing Maps Improved by Asymmetric Neighborhood Function. Cognitive neurodynamics, 3(1), pp. 9-15.

[3] Brownlee, J. (2011). Clever algorithms: nature-inspired programming recipes. Jason Brownlee.

[4] Chaudhary, V., Ahlawat, A. K. and Bhatia, R. S. (2011). Growing Neural Networks using Soft Competitive Learning. International Journal of Computer Applications, pp. 975-987.

[5] Diehl, P. U. and Cook, M. (2015). Unsupervised Learning of Digit Recognition using Spike-Timing-Dependent Plasticity. Frontiers in Computational Neuroscience, 9.

[6] Erwin, E., Obermayer, K., and Schulten, K. (1992). Self-Organising Maps: Ordering, Convergence Properties and Energy Functions. Biological Cybernetics, 67: 47-55. DOI: 10.1007/BF00201801.

[7] Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. Annals of human genetics, 7(2), pp.179-188.

[8] Gurney, K. (2004). An Introduction to Neural Networks. London: Taylo and Francis e-Library, University College London (UCL).

[9] Kaiichiro, O., Takaaki, A., Koji, K and Toshio, A. (2011). Asymmetric Neighborhood Functions Accelerate Ordering Process of Self-Organizing Maps. Physical Review. E, Statistical, Nonlinear, and Soft Matter Physics Journal Vol 83, DOI: 10.1103/PhysRevE.83.021903.

[10] Kohonen, T. (1982). Self-Organized Formation of Topologically Correct Feature Maps. Biological Cybernetics, 43, pp. 59-69.

[11] Madani, K. (2006). Industrial and Real World Applications of Artificial Neural Networks: Illusion or reality? Informatics in Control, Automation and Robotics, Vol. I, Springer, ISBN 1-4020-4136-5, pp. 11-26.

[12] Malsburg, C. (1973). Self-Organization of Orientation Sensitive Cells in the Striate Cortex. Kybernetic, 14, pp. 85-100.

[13] Salomon, L. A., Fort, J. C. and Chang, L. V. (2012). Average Competitive Learning Vector Quantization. MAP5 2012-10 <hal-00685960>.

[14] Schmidhuber, J. (2015). Deep Learning in Neural Networks: An Overview. Neural Networks, Vol 61, pp 85-117. DOI: 10.1016/j.neunet.2014.09.003.

[15] Yang, X. S. (2010). A New Metaheuristic Bat-Inspired Algorithm. In: Nature Inspired Cooperative Strategies for Optimisation, pp. 65-74. Springer.

[16] Yang, X. S. and Gandomi, A. H. (2010). Bat Algorithm: A Novel Approach for Global Engineering Optimization. Vol. 29 Iss: 5 pp. 464 –483.