An Efficient Neural Gas Network for Classification

A.S. Atukorale and P.N. Suganthan
Department of Computer Science and Electrical Engineering
University of Queensland, St Lucia, QLD 4072, Australia.
{ajantha,sugan}@csee.uq.edu.au,

Abstract In this paper, we describe our investigations into the neural gas network. The original neural gas network is computationally expensive, as an explicit ordering of all distances between synaptic weights and the training sample is necessary. This has a time complexity of $O(NlogN)$ in a sequential implementation. We propose an alternative scheme in which the ordering is carried out implicitly. In addition, to further speed-up the sequential implementation, we used a truncated exponential function as its neighbourhood function as in [1]. We also developed a hierarchical overlapped neural gas architecture for pattern classification. This allowed us to obtain multiple classifiers and the final classification was made by fusing the individual classifiers.

1 Introduction

Interest in neural networks, machine learning and parallel computation has aroused a renewed enthusiasm in the area of statistical pattern recognition [2]. Adaptive non-parametric neural-net classifiers, especially unsupervised exemplar classifiers, work well for many real-world problems. The continuing increase in the availability of computing speed and especially storage capacity has facilitated the application of powerful learning algorithms towards achieving this success.

Neural network models have been intensively studied for many years in an effort to obtain human-like performance. The self-organising feature map (SOFM) proposed by Kohonen [3, 4] is one of the network paradigm widely used for solving complex problems such as vector quantisation, speech recognition, combinatorial optimisation, pattern recognition and modelling the structure of the visual cortex. Kohonen's feature map is a special way for conserving the topological relationships in input data, which also bears some limitations. In this model the neighbourhood relations between neural units have to be defined in advance. The topology of the input space has to match the topology of the output space which is to be represented. In addition, there exist no cost function that yields Kohonon's adaptation rule as its gradient. Some of the above issues have been discussed in the literature by Blackmore et al. [5], Kangas et al. [6], Li et al. [7], Bauer et al. [8] and Fritzke [9].

Martinetz et al. [10, 11] proposed the neural gas (NG) network algorithm for vector quantisation, prediction and topology representation a few years ago. The NG network model 1) converges quickly to low distortion errors, 2) reaches a distortion error lower than that resulting from K-means clustering, maximum-entropy clustering and Kohonen's SOFM, 3) obeys a gradient descent on an energy surface. Similar to the SOFM algorithm the NG algorithm uses a *soft-max* adaptation rule (ie. not only adjust the *winner* reference vector, but also affects all the cluster centers depending on their proximity to the input signal) to avoid confinement to local minima during the adaptation procedure. Despite all those advantages, the NG network suffers from high computational complexity for its sequential implementation [1].

In this paper, we discuss how above computational complexity problem can be reduced. In addition, by defining an hierarchical overlapped structure [12] on top of the standard NG network, we obtain a hierarchical overlapped neural gas (HONG) model for the classification of real world handwritten digits with high variations.

The paper is organised as follows. After a brief review of the NG network algorithm in Section 2, we present our proposed speed-up technique in Section 3. In section 4 we present the HONG architecture and in Section 5, we present the experimental results. The paper is concluded with a brief discussion in Section 6.

2 The Neural Gas Algorithm

In this approach the synaptic weights $\boldsymbol{w_i}$ are adapted independently of any topological arrangement of the neural units within the network. Instead, it utilises a *neighbourhood-ranking* of the synaptic weights $\boldsymbol{w_i}$ for a given data vector $\boldsymbol{v}$. The synaptic weight changes $\Delta \boldsymbol{w_i}$ are not determined by the relative distances between the neural units within a topologically prestructured lattice, but by the relative distances between the neural units within the input space: hence the name *neural gas* network.

Information about the arrangement of the receptive fields within the input space is implicitly given by set of distortions $D_v = \{\|\boldsymbol{v} - \boldsymbol{w_i}\|, i = 1, ..., N\}$ associated with each $\boldsymbol{v}$, where N is the number of units in the network [10]. Each time an input signal $\boldsymbol{v}$ is presented, the ordering of the elements of the set D_v is necessary to determine the adjustment of the synaptic weights $\boldsymbol{w_i}$. This ordering has a time complexity of $O(NlogN)$ in sequential implementation. The resulting adaptation rule can be described as a *winner-take-most* instead of *winner-take-all* rule.

A presented input signal $\boldsymbol{v}$ is received by each neural unit i and induces *excitations* $f_i(D_v)$ which depend on the set of distortions D_v. Assuming a Hebb-like rule,

$$\Delta \boldsymbol{w_i} = \epsilon \cdot f_i(D_v) \cdot (\boldsymbol{v} - \boldsymbol{w_i}) \tag{1}$$

The step size $\epsilon \in [0,1]$ describes the overall extent of the modification (learning rate) and $f_i(D_v) \in [0,1]$ accounts for the topological arrangement of the $\boldsymbol{w_i}$ within the input space. Martinetz et al. [10] reported that an exponential function $\exp(-k_i/\lambda)$ should give the best overall result, compared to other choices like Gaussians for the *excitation* function $f_i(D_v)$, where λ determines the number of neural units significantly changing their synaptic weights with the adaptation step (1). The rank index $k_i = 0, ..., (N-1)$, describes the *neighbourhood-ranking* of the neural units, where $k_i = 0$ being the closest synaptic weight ($\boldsymbol{w_{i_0}}$) to $\boldsymbol{v}$, $k_i = 1$ being the second closest synaptic weight ($\boldsymbol{w_{i_1}}$) to $\boldsymbol{v}$, and so on. That is, the set $\{\boldsymbol{w_{i_0}}, \boldsymbol{w_{i_1}}, ..., \boldsymbol{w_{i_{N-1}}}\}$ is the *neighbourhood-ranking* relative to the given input vector $\boldsymbol{v}$. The *neighbourhood-rank* index k_i depends on $\boldsymbol{v}$ and the whole set of synaptic weights $\boldsymbol{W}=\{\boldsymbol{w_1}, \boldsymbol{w_2}, ..., \boldsymbol{w_N}\}$ and we denote this as $k_i(\boldsymbol{v}, \boldsymbol{W})$. The NG network algorithm is summarised in Table 1.

Table 1: The Neural Gas Algorithm

NG1: Initialise synaptic weights $\boldsymbol{w_i}$, $i = 1, ..., N$ and training parameters (λ_i, λ_f, ϵ_i, ϵ_f) where λ_i, ϵ_i are initial values of $\lambda(t), \epsilon(t)$ and λ_f, ϵ_f are final values of $\lambda(t), \epsilon(t)$.

NG2: Present an input vector $\boldsymbol{v}$ and compute distortions.

NG3: Order the distortion set D_v in ascending order.

NG4: Adapt weight vectors according to:

$$\Delta \boldsymbol{w_i} = \epsilon(t) \cdot h_\lambda(k_i(\boldsymbol{v}, \boldsymbol{W})) \cdot (\boldsymbol{v} - \boldsymbol{w_i}) \tag{2}$$

where $i = 1, ..., N$. The parameters have the following time dependencies: $\lambda(t) = \lambda_i(\lambda_f/\lambda_i)^{t/t_{max}}$, $\epsilon(t) = \epsilon_i(\epsilon_f/\epsilon_i)^{t/t_{max}}$, $h_\lambda(k_i) = exp(-k_i/\lambda(t))$.

NG5: Increment the time parameter t by 1.

NG6: Repeat NG2 - NG5 until the maximum iteration t_{max} is reached.

3 Implicit Ranking Scheme

Recently some work has been done on speed-up procedures for the sequential implementation of the NG algorithm. Ancona et al. [13] discussed the questions of sorting accuracy and sorting completeness. With theoretical analysis and experimental evidence they have concluded that partial, exact sorting (ie. ordering top few *winning* units correctly and keeping all other units in the list unaffected) performs better than complete but noisy sorting (ie. ordering top few *winning* units correctly and all other remaining units are subjected to inexact sorting) and even a few units in partial sorting is sufficient to attain a final distortion equivalent to that attained by original NG algorithm. Moreover, they have concluded that correct identification of the best-matching unit becomes more and more important while training proceeds. This is obvious, as training proceeds, the adaptation step (1) becomes equivalent to the K-means adaptation rule. Choy et al. [1] applied partial distance elimination (PDE) method to speed-up the NG algorithm in the above context.

In our investigations, we eliminate the explicit ordering (see Table 1, **NG3**) by employing the following implicit ordering metric:

$$m_i = \frac{d_i - d_{min}}{d_{max} - d_{min}} \tag{3}$$

where d_{min} and d_{max} are the minimum and maximum distances between the training sample and all reference units in the network respectively and $d_i \in D_v$, $i = 1, ..., N$. The best matching *winner* unit will then have an index of 0, the worst matching unit will have an index of 1 and other units will take values between 0 and 1. ie., $m_i \in [0, 1]$.

By employing the above modification to the original NG algorithm, we can modify the two entries **NG3** and **NG4** in Table 1 as follows.

Table 2: Modified Neural Gas Algorithm

NG3: Find d_{min}, d_{max} from the distortion set D_v.

NG4: Adapt weight vectors according to:

$$\Delta \boldsymbol{w_i} = \epsilon(t) \cdot h_{\lambda'}(m_i(\boldsymbol{v}, \boldsymbol{W})) \cdot (\boldsymbol{v} - \boldsymbol{w_i}) \tag{4}$$

where $h_{\lambda'}(m_i) = exp(-m_i/\lambda'(t))$ and $\lambda'(t) = \lambda(t)/(N-1)$.

In addition, we also updated only those units with a non-negligible effective learning rate as in [10, 1]. This is implemented by a further modification of the weight updating phase with the following truncated exponential function as in [1].

$$h_\lambda(m_i) = \begin{cases} \exp(-m_i/\lambda(t)) & \text{if } m_i < r \cdot \lambda'(t) \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where r is a constant to be selected. Because of the above truncation, the weight update rule (4) will update those weights with non-zero value of $h_\lambda(m_i)$. These modifications were able to eliminate the ranking mechanism completely and also reduce the number of updates substantially - by almost 80% on average.

4 Hierarchical Overlapped Architecture

By retaining the essence of the original NG algorithm and our modification, we developed a hierarchical overlapped neural gas (HONG) network algorithm for labeled pattern recognition.

The structure of the HONG architecture is as follows. First, the network is initialised with just one layer which is called the base layer. The number of units in the base layer has to be chosen appropriately. In labeled pattern recognition applications, the number of distinct classes and the number of training samples may be considered in the selection of the initial size of the network. In this architecture every unit has a synaptic weight vector which has the same dimension as the input feature vector. Once we have selected the number of units in the base layer, we applied our modified version of the NG algorithm to adapt the synaptic weights of the units in the network. Having completed the unsupervised NG learning, the units in the base layer were labeled using a simple voting mechanism. In order to fine tune the network, we applied the supervised learning vector quantisation (LVQ) [3] algorithm to the labeled network. Then we obtained the overlaps for each unit in the base layer. The overlapping is achieved by duplicating every training sample to train several upper level NGs. That is, the winning unit as well as a number of runners-up units make use of the same training sample to train the upper level NG networks grown from those units in the base layer NG. For example, in figure 1, the overlapped NG grown from unit **A** is trained on samples where the unit **A** is a winner or one of the first few runners-up for all the training samples presented to the base layer at the end of the training phase. In this way, we obtain several upper level NGs which have the same training sample to train each of them. If we consider the *winning* unit as well as 10 runners-up units for each training sample in the base NG, then we obtain 11 upper level NGs for each sample. Figure 1 also shows the overlap in the feature space of the two overlapped NGs conceptually.

The testing samples are also duplicated, but to a lesser degree. Hence the testing samples fit well inside the feature space spanned by the *winner* and several runners-up in the training data. In addition, this duplication of the samples allows us to employ one of the several classification schemes to obtain the final classification [14, 15].

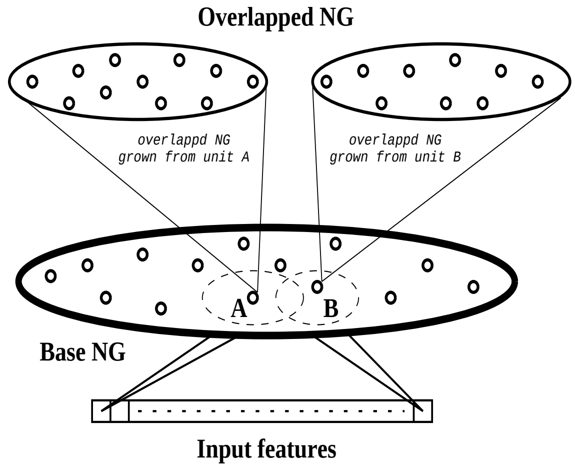

Figure 1: Hierarchical overlapped architecture

In combining classifiers, first we employed majority voting scheme and weighted voting scheme. Then we extended our study to obtain a confidence value for every sample for its *membership* in every class(j) using the following:

$$C_j = \frac{d_{min}}{d_{min\_label\_j}} \qquad (6)$$

where $j = 0, ..., 9$. This will define the ratio between the global minimum distance (d_{min}) and minimum distance for every class ($d_{min\_label\_j}$) as the confidence value for the input pattern to belong to the j^{th} class. Naturally, the class which has the global minimum distance will yield a confidence value of 1. These confidence values generated by multiple NG classifiers were accumulated and the label with the highest aggregate was chosen as the final classification decision. The HONG network algorithm is summarised in Tabel 3.

Table 3: The HONG network Algorithm

HONG1: Initialise the synaptic weights and training parameters of the base NG.

HONG2: Train the base NG using the modified NG algorithm as explained in (4).

HONG3: Label the base NG using simple voting scheme.

HONG4: Fine tune the base map with the supervised LVQ algorithm.

HONG5: Obtain the overlaps for each unit in the base layer.

HONG6: Initialise the synaptic weights of the second level NGs around its base layer's unit (root unit).

HONG7: Train each second layer NG networks as in HONG2, Label each of them accordingly as in HONG3 and fine tune them as in HONG4.

HONG8: Obtain the final training and testing rates by combining the results of the overlapped NGs in the second layer.

5 Experimental Results

We performed experiments on handwritten digits to test our proposed HONG network. The handwritten digit samples were extracted from the NIST 19 database, from files by_class/30/hsf_0.mis to by_class/39/hsf_0.mis. The feature extraction algorithm is as same as the one used by Suganthan [12]. The algorithm is summerised as follows:

- Each character is pre-processed. This involves removing of isolated blobs from the binary image.
- Each character is centered and rescaled from 128x128 to 88x72 pixel resolution.
- Each scaled binary image is divided into 8x8 blocks and the result is an 11x9 grey scale image with pixel values in the range of [0,64].

As a result of pre-processing, we were left with a feature vector of 99 elements. A few samples are shown in Figure 2. Further, the extracted samples were divided into three sets: namely training, validation and testing. We restricted the number of upper level layers to two. The first layer consisted of 200 units. The number of units for each NG in the second layer were determined empirically by considering the available training samples for each of them. We found that $max\{35, (training\_samples)/8\}$ is a good estimate in determining the number of units for the second layer NGs.

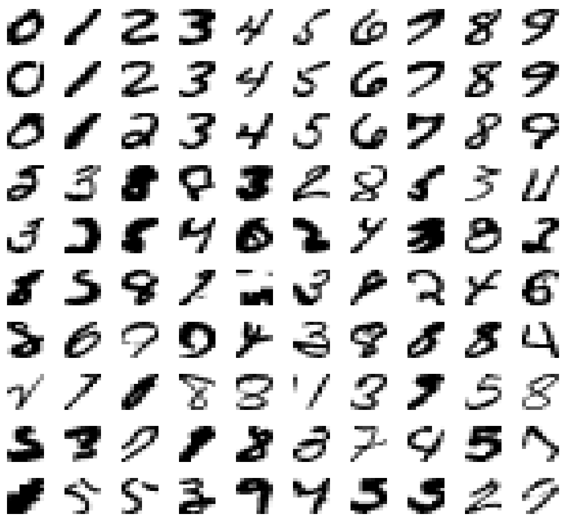

Figure 2: Some of the samples used.

We used 11 overlaps for the training set and 7 overlaps for the test set. We also used other combinations such as 13-9, 7-4 and so on. But 11-7 is a good compromise between the accuracy and the training time. To truncate the exponential function (5), a value between 11 and 13 was a good approximation for the parameter r. We chose $\epsilon_i = 0.7, \epsilon_f = 0.05, \lambda_i = 0.1, \lambda_f = 0.0001$ and $t_{max} = 92,000$. The data set we used had 23,000 training samples, 11,500 validation samples and 11,500 testing samples. The recognition rates obtained using the above parameters are illustrated in Table 4. As can be seen, the HONG architecture improves further on the high classificaton rate provided by the base NG network.

Table 4: Recognition Rates

	Training	Validation	Testing
Base Map	99.06%	98.03%	97.78%
HONG	99.81%	99.03%	99.03%

6 Conclusions

In this paper, we proposed an implicit ranking scheme to speed-up the sequential implementation of the original NG algorithm. In contrast to Kohonen's SOFM algorithm, the NG algorithm takes smaller number of learning steps to converge, does not require any prior knowledge about the structure of the network and at the same time it minimises the global cost function.

By making the NG algorithm run faster, we developed the HONG network architecture to obtain a better classification on conflicting data (resulting in a condition known as *gridlock* [16]). The HONG network architecture successively partition the input space to avoid such situations (see Figure 1). Since the training and the testing samples are duplicated in the upper layers, we obtain multiple decision classifications for every sample. The final classification is obtained by fusing the individual classifications. We employed the idea of confidence values in obtaining the final classification. The proposed architecture was tested on handwritten numerals extracted from the NIST 19 database.

Compared to the number of applications in Kohonen's SOFM, there are relatively few for NG in the literature [17, 18, 19, 20, 21, 22]. We hope, due to the speedup method that we have introduced for the sequential implementation, there will be more applications in NG algorithm in the future.

References

[1] C. S.-T. Choy and W.-C. Siu, "Fast Sequential Implementation of Neural Gas Network for Vector Quantization," *IEEE Transactions on Communications*, vol. 46, pp. 301–304, Mar. 1998.

[2] R. P. Lippmann, "Pattern Classification using Neural Networks," *IEEE Communications Magazine*, vol. 11, pp. 47–64, Nov. 1989.

[3] T. Kohonen, "The Self-Organizing Map," *Proceedings of the IEEE*, vol. 78, pp. 1464–1480, Sept. 1990.

[4] T. Kohonen, "Self-Organized Formation of Topologically Correct Feature Maps," *Biological Cybernetics*, vol. 43, pp. 59–69, 1982.

[5] J. Blackmore and R. Miikkulainen, "Incremental grid growing: Encoding high-dimensional structure into a two-dimensional feature map," Tech. Rep. AI92-192, University of Texas at Austin, Dec. 1992.

[6] J. A. Kangas, T. K. Kohonen, and J. T. Laaksonen, "Variants of Self-Organizing Maps," *IEEE Transactions on Neural Networks*, vol. 1, pp. 93–99, Mar. 1990.

[7] T. Li, Y. Y. Tang, and L. Fang, "A Structure-Parameter-Adaptive (SPA) Neural Tree for the Recognition of Large Character Set," *Pattern Recognition*, vol. 28, no. 3, pp. 315–329, 1995.

[8] H.-U. Bauer and T. Villmann, "Growing a Hypercubical Output Space in a Self-Organizing Feature Map," *IEEE Transactions on Neural Networks*, vol. 8, pp. 218–226, Mar. 1997.

[9] B. Fritzke, "Growing Cell Structures - A Self-Organizing Network for Unsupervised and Supervised Learning," *Neural Networks*, vol. 7, no. 9, pp. 1441–1460, 1994.

[10] T. Martinetz and K. Schulten, "A Neural-Gas Network Learns Topologies," in *Artificial Neural Networks* (T. Kohonen, K. Mäkisara, O. Simula, and J. Kangas, eds.), pp. 397–402, Amsterdam: North-Holland, 1991.

[11] T. Martinetz, S. G. Berkovich, and K. Schulten, "Neural Gas Network for Vector Quantization and its Application to Time-Series Prediction," *IEEE Transactions on Neural Networks*, vol. 4, pp. 218–226, July 1993.

[12] P. N. Suganthan, "Structure Adaptive Multilayer Overlapped SOMs with Supervision for Handprinted Digit Classification," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN'98)*, pp. 1706–1711, Anchorage, Alaska, USA, May 1998.

[13] F. Ancona, S. Ridella, S. Rovetta, and R. Zunino, "On the Importance of Sorting in Neural Gas Training of Vector Quantizers," in *Proc. of the IEEE International Conference on Neural Networks*, pp. 1804–1808, 1997.

[14] J. Kittler, "Combining Classifiers: A Theoretical Framework," *Pattern Analysis and Applications*, vol. 1, no. 1, pp. 18–27, 1998.

[15] S.-B. Cho and J. H. Kim, "Multiple Network Fusion using Fuzzy Logic," *IEEE Transactions on Neural Networks*, vol. 6, pp. 497–501, Mar. 1995.

[16] H. Ranganath, D. Kerstetter, and S. Sims, "Self Partitioning Neural Networks for Target Recognition," *Neural Networks*, vol. 8, no. 9, pp. 1475–1486, 1995.

[17] B. Zhang, M. Fu, and H. Yan, "Handwritten Digit Recognition by Neural Gas Model and Population Decoding," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN'98)*, pp. 1727–1731, Anchorage, Alaska, USA, May 1998.

[18] M. Fontana, N. Borghese, and S. Ferrari, "Image Reconstruction using Improved Neural Gas," in *Italian Workshop on Neural Nets (7th), Vietri Sul Mare, Italy* (M. M. et al., ed.), pp. 260–265, 1996.

[19] T. Hofmann and J. M. Buhmann, "An Annealed Neural Gas Network for Robust Vector Quantization," in *Artifitial Neural Networks (ICANN '96)* (C. von der Malsburg, W. von Seelen, J. Vorbruggen, and B. Sendhoff, eds.), vol. 7, pp. 151–156, Bochum Germany: Springer, July 1996.

[20] E. Ardizzone, A. Chella, and R. Rizzo, "Color Image Segmentation Based on a Neural Gas Network," in *International Conference of Artifitial Neural Networks (ICANN '94), Sorrento, Italy* (M. M. et al., ed.), pp. 1161–1164, May 1994.

[21] K. Kishida, H. Miyajima, and M. Maeda, "Destructive Fuzzy Modeling Using Neural Gas Network," *IEICE Transactions on Fundermentals of Electronics, Communications and Computer Sciences*, vol. E80-A, pp. 1578–1584, Sept. 1997.

[22] B. Zhang, M. Fu, and H. Yan, "Application of Neural Gas Model in Image Compression," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN'98)*, pp. 918–921, Anchorage, Alaska, USA, May 1998.