Double Growing Neural Gas for Disease Diagnosis

Guojian Cheng, Andreas Zell
University Tübingen
Wilhelm-Schickard-Institut for Computer Science
Dept. Computer Architecture
Köstlinstr. 6, D-72074 Tübingen, Germany
{cheng,zell}@informatik.uni-tuebingen.de

Abstract

We present here a variant of B. Fritzke's GNG (Growing Neural Gas) — Double GNG (DGNG). In each insertion step of the GNG only one new cell is inserted in the middle of the edge connecting Maximum Resource Vertex (MRV) and a MRV in its direct topological neighbourhood. But in our DGNG two new cells are inserted at the same time. Our goal is to speed up the convergence of the learning process. Although multiple growing cell mechanism can reduce the required number of learning epochs, it can also lead to an increased network structure. Simulation results on some neural network benchmarks indicate that, for many data sets, when the number of new cells in each insertion step is within three, the total performance of networks is at best, but that increasing the number of new cells beyond three has very little benefit and sometimes degrades performance. In this paper we consider only the DGNG. With two disease diagnosis benchmarks (Wisconsin breast cancer and soybean disease diagnosis) we tested the DGNG and indicated that DGNG performs better than the original GNG, measured by the required number of epochs and CPU times.

1 Introduction

Kohonen's Self-Organizing Maps (SOM) [4] generate mappings from high dimensional signal spaces to lower dimensional topological structures. Their main features are formation of topology-preserving feature maps and approximation of the input probability distribution. However, SOMs have some disadvantages: their fixed number of neural units makes them impractical for applications where the optimal number of units is not known in advance. Also, a topology of fixed dimensionality results in problems if this dimensionality does not match the dimensionality of the feature manifold. Incremental artificial neural networks grow as they learn and shrink as they forget. Based on SOM and Competitive Hebbian Learning (CHL) [5], B. Fritzke introduced an incremental self-organizing network with variable topology, known as Growing Neural Gas (GNG) [1]. The GNG requires no determination of the network dimensionality in advance. Instead the algorithm starts with two units at random positions and inserts new nodes at positions with high errors or resource values (the accumulated errors play a nutrition-like role for the cell growing). These nodes are connected by edges with a certain age, old edges are removed, which results in a topology preserving map with an induced Delaunay triangulation. All parameters used by GNG are constant. It is, therefore, not necessary to define a decay schedule as for the SOM.

Through a combination of GNG with radial basis functions (RBF) the Supervised GNG (SGNG) was developed [2]. The SGNG has three main advantages over other traditional supervised neural networks: First, number, diameter and position of RBF units are determined automatically through the growth process, which can be stopped as soon as the network performance is good enough. Second, since positioning of RBF units and supervised training of connection weights is performed in parallel, the current classification error (used as resource value) can be used to determine where to insert new RBF units. Third, the networks can generalize well and have a relatively small size. A performance comparisons of GNG with other neural network models have been made by M. Kunze [7] and F. Hamker [3].

2 Double Growing Neural Gas

2.1 The Mechanism of DGNG

The cell growing mechanism of DGNG is nearly the same as in GNG. The only difference is that in DGNG two new cells are inserted simultaneously instead of only one new cell as in GNG. In other words, in each insertion step of GNG only one new cell is inserted in the middle of the edge connecting the Maximum Resource Vertex (MRV) and it's maximum source neighbour node. But in DGNG two new cells are inserted at the same time. The first new cell is inserted in the same way as in GNG. In addition, another new cell is inserted between the edge connecting the second MRV and it's maximum source neighbour node. When the second MRV is in the direct topological neighbourhood of the first MRV, we insert only one new cell between first MRV and second MRV. It is important that the new cell should be inserted in such way that the expected value of a certain error measure (resource value) in the following process becomes equal for all neurons.

2.2 The Algorithm of Supervised DGNG

The algorithm diagram of supervised DGNG is showed on the next page (Table 1).

3 Disease Diagnosis Simulation Results

We applied our supervised DGNG algorithm to two neural network disease diagnosis benchmarks, Wisconsin breast cancer study and soybean disease recognition problem, which are adopted from the PROBEN1[6]. For the application of the validation method the datasets were divided into three sets: training set, validation set and test set. For controlling the training process the early stopping method was used. The criterion can be expressed as: stop training as soon as the error on the validation set is higher than it was the last time it was checked.

Table 1: Supervised DGNG Algorithm

- Start with a random edge of 2 connected cells at age zero.
- Initialize the learning parameters: λ, ε_{win}, ε_{dtn}, η, α, β, θ.
- Create m linear output units and a weighted connection $\mathbf{w}_{jc}$ from each competitive cell c to each output unit $j, (j \in \{1, 2, \ldots, m\})$.
- Associate every competitive cell with a Gaussian function.
- while (classification error is not zero or maximum epoch is not reached):
  - λ-Times Adaptation:
    - Load I/O-pattern pair (ξ, ζ) from training data.
    - Determine the first winner C_{win_1} and the second C_{win_2}.
    - Are C_{win_1} and C_{win_2} connected by an edge ?
Yes: reset the corresponding edge-age to zero;
No: connect C_{win_1} with C_{win_2}.
    - Increment the age of all edges emanating from C_{win_1}.
    - Remove edges at a certain age θ as well as resulting non-connected cells.
    - Increase matching for C_{win_1} and its direct topological neighbour cells by learning rates ε_{win} and ε_{dtn} resp. ($\varepsilon_{win} \gg \varepsilon_{dtn}$).
    - Compute activation d_c for every competitive cell c according to the Gaussian function.
    - All output unit activations are calculated by the product of $\mathbf{w}_{jc}$ and d_c.
    - Perform one δ-rule step for the output weights by learning rate η.
    - Increase resource variable R_{win1}.
  - Determine the cell C_{max1} and C_{max2} with the first and second maximum resource resp.
  - Determine the cell C'_{max2} with the maximum resource in the direct topological neighbourhood of C_{max1}.
  - Determine the cell C_{mrn1} and C_{mrn2} with the maximum resource of direct topological neighbourhood of C_{max1} and C_{max2} resp.
  - $C'_{max2} = C_{max2}$?
Yes: insert a new cell C_{new1} between C_{max1} and C_{max2}.
No: insert the first new cell C_{new1} between C_{max1} and C_{mrn1}; then insert the second new cell C_{new2} between C_{max2} and C_{mrn2}.
  - Decrease the resource values of $C_{max1,2}$ and $C_{mrn1,2}$ by a fraction α.
  - Interpolate the resource variable and output weight vector of the new cells C_{new1} and C_{new2}.
  - Decrease the resource variable of all units by a factor β.

3.1 Wisconsin Breast Cancer Study

The breast cancer diagnosis is derived from the descriptions gathered by microscopic examination. These descriptions include the clump thickness, the uniformity of cell size and cell shape, the amount of marginal adhesion, and the frequency of bare nuclei. The dataset consists of 9 inputs and 699 examples. The training set, validation set and test set contained 350, 175 and 174 patterns, respectively. For this simulation, the two classes (benign or malignant tumor) were coded on one outputs neuron. The accumulated squared output error served as resource updating: $\Delta R_{win} = ||\zeta - \mathbf{o}||^2, \zeta, \mathbf{o} \in \{0,1\}$.

Table 2: Comparison DGNG with SGNG for the Wisconsin cancer classification problem.

Database	Algorithm	Epochs	CPU Time (Seconds)	Training Set CR	Testing Set CR
Cancer1	SGNG	154	42.06	100%	96.55%
Cancer1	DGNG	96	28.02	100%	97.70%
Cancer2	SGNG	156	40.98	100%	95.98%
Cancer2	DGNG	93	27.64	100%	96.55%
Cancer3	SGNG	146	36.82	100%	98.28%
Cancer3	DGNG	80	21.77	100%	97.13%

Table 2 shows the difference between SGNG and DGNG for this problem. CR represents the Classification Rate. The IBM-RS/6000 computer was used for testing of CPU time. As it can be seen the number of epochs and CPU time required by the DGNG is between 45% and 33% smaller than for original SGCS, while the generalization performance for SGCS and DGNG is almost same.

3.2 Soybean Disease Recognition Problem

This benchmark problem concerns the recognition of 19 different diseases of soybeans. The discrimination is done based on a description of the bean (e.g. whether its size and color are normal) and the plant (e. g. the size of spots on the leafs, whether these spots have a halo, whether plant growth is normal, whether roots are rotted) plus information about the history of the plant's life (e. g. whether changes in crop occurred in the last year or last two years, whether seeds were treated, how the environment temperature is).

The soybean benchmark dataset consists of 82 inputs, 19 outputs and 683 samples. This is the problem with the highest number of classes in the PROBEN1 benchmark set. Most attributes have a significant number of missing values. For this simulation, the 19 classes were coded on 19 outputs neurons, each class being associated with a neuron. The accumulated squared output error served as resource updating: $\Delta R_{win} = ||\zeta - \mathbf{o}||^2, \zeta, \mathbf{o} \in \{0,1\}$. The soybean dataset with 342 examples used for the training set, 171 examples used for the validation set. 170 examples used for the testing set.

Table 3 shows the difference between SGNG and DGNG for this problem. As it can be seen the number of epochs and CPU time required by the DGNG is between

Table 3: Comparison of DGNG and SGNG for the soybean disease recognition problem.

Database	Algorithm	Epochs	CPU Time (Seconds)	Training Set CR	Testing Set CR
Soybean1	SGNG	125	132.55	94.44%	89.41%
Soybean1	DGNG	70	65.95	94.44%	89.41%
Soybean2	SGNG	105	107.75	90.94%	82.35%
Soybean2	DGNG	70	74.40	91.81%	84.12%
Soybean3	SGNG	120	129.50	93.27%	90.00%
Soybean3	DGNG	80	99.29	92.40%	87.64%

50% and 23% smaller than for original SGCS. For soybean1, to achieve same classification rate on testing set SGNG takes 125 epochs and 132.55 CPU seconds, but DGNG takes only 70 epochs and 65.95 seconds. For soybean2 the generalization performance of DGNG is better than SGNG. The classification rate is increased from 82.35% (SGNG) to 84.12% (DGNG). For soybean3 the generalization performance of DGNG is not so good as of SGNG, but DGNG needs only 80 epochs and 99.29 seconds CPU time.

4 Conclusions

In this paper we have developed a novel variant of Fritzke's GNG network. The main advantage of GNG is that it can automatically find a network structure and size suitable for a given classification pattern through cell growing. The DGNG algorithm can speed up the convergence of the learning process. Comparative experimental results on two benchmark datasets revealed that DGNG is both effective and efficient as a learning algorithm for diesease diagnosis problems. The required epochs in DGNG are about half to two third of the epochs required by GNG. The number of cells required for DGNG and GNG are nearly equal.

Acknowledgments

We would like to thank B. Fritzke, M. Kunze, F. Hamker, J. Bruske and L. Prechelt for valuable and constructive discussions about their growing neural network algorithm and neural network benchmark problems. We also wish to thank our colleagues Igor Fischer, Jürgen Wakunda, Fred Rapp and Jutta Huhse for their help in programming and proof reading. The first author also thanks German KAAD (Katholischer Akademischer Ausländer-Dienst) very much for the scholarship.

References

[1] B. Fritzke. A growing neural gas network learns topologies. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 625–632. MIT Press, Cambridge MA, 1995.

[2] B. Fritzke. Automatic construction of radial basis function networks with the growing neural gas model and its relevance for fuzzy logic. In *Applied Computing 1996: Proceedings of the 1996 ACM Symposium on applied computing*, pages 624–627, Philadelphia, 1996. ACM.

[3] F. Hamker and D. Heinke. Implementation and comparison of growing neural gas, growing cell structures and fuzzy artmap. In *Technical report, Schriftenreihe des FG Neuroinformatik der TU Ilmenau*. Report 1/97, ISSN 0945-7518, 1997.

[4] Teuvo Kohonen. *Self-Organizing Maps*. Springer, Berlin, Heidelberg, 1995. (Second Extended Edition 1997).

[5] Thomas Martinetz. Competitive hebbian learning rule forms perfectly topology preserving maps. In Stan Gielen and Bert Kappen, editors, *Proc. ICANN'93, Int. Conf. on Artificial Neural Networks*, pages 427–434, London, UK, 1993. Springer.

[6] Lutz Prechelt. PROBEN1 — A set of benchmarks and benchmarking rules for neural network training algorithms. Technical report, Fakultät für Informatik, Universität Karlsruhe, Germany.

[7] M. Kunze R. Berlich and J. Steffens. A comparison between the performance of feed forward neural networks and the supervised growing neural gas algorithm. In *In 5th Artificial Intelligence in High Energy Physics workshop, Lausanne*, 1996.