# Comparing Neural Networks: A Benchmark on Growing Neural Gas, Growing Cell Structures, and Fuzzy ARTMAP

Dietmar Heinke and Fred H. Hamker

*Abstract*— **This article compares the performance of some recently developed incremental neural networks with the well-known multilayer perceptron (MLP) on real-world data. The incremental networks are fuzzy ARTMAP (FAM), growing neural gas (GNG) and growing cell structures (GCS). The real-world datasets consist of four different datasets posing different challenges to the networks in terms of complexity of decision boundaries, overlapping between classes, and size of the datasets. The performance of the networks on the datasets is reported with respect to measure classification error, number of training epochs, and sensitivity toward variation of parameters. Statistical evaluations are applied to examine the significance of the results. The overall performance ranks in the following descending order: GNG, GCS, MLP, FAM.**

*Index Terms*— **Benchmark, comparison of neural networks, fuzzy ARTMAP (FAM), growing neural gas (GNG), growing cell structures (GCS), multilayer perceptron (MLP), real-world data.**

## I. Introduction

RECENTLY, the number of neural network paradigms has increased dramatically. This development led us to the question of which is the "best" neural network for solving a pattern classification task. In the present paper we consider this question within the following framework: First, the answer should be given for a real-world application. Thus, the pattern classification tasks for this benchmark should use real-world data which are widely available, so that results are reproducible and build a foundation for the evaluation of new networks. The datasets should comprise different properties posing different challenges to the networks.

Second, the results should yield some general statements about the performance of the neural networks. Since such a benchmark produces empirical results, statistical evaluations are necessary to examine the relevance of the results [1]. If this does not lead to a clear, general answer, at least some rules, e.g., properties of the data, should be stated under which certain performances are to be expected.

Third, our tested neural networks should in principle be able to learn new patterns without forgetting the old ones. This goal is often discussed in the literature under terms such as "lifelong learning," "on-line learning," or "incremental learning" [2]. This aspect comes from the objective of the MIRIS project within which framework this research was done [3], [4]. The goal of this project is a robot-vision system that operates under changing environmental conditions and changing physical characteristics of nonuniform textured objects. In our opinion, incremental networks like the growing neural gas (GNG), growing cell structures (GCS) and fuzzy ARTMAP (FAM) are good candidates for this task. Instead of only testing these networks on our own datasets, we decided to compare them also on a public dataset.

The need for such a benchmark using real-world statistical evaluations and comparison of different neural networks has been recently highlighted by a number of researchers ([5], [1], [6]). However, there already exist some benchmarks: For example, a relatively new benchmark and collection of software at the University of Toronto is DELVE [7]. DELVE aims at giving researchers the possibility to compare their approaches with others on many datasets.

An impressive benchmark is also provided by the ELENA-report [8]. It considers seven classifiers in connection with three artificial databases and four real-world databases. However, the only neural network it considers is the multilayer perceptron (MLP), which is compared with classifiers such as learning vector quantization, Gaussian quadratic classifiers, and others. In most cases MLP achieves good results. Hence, a better classifier than MLP might also be better than the classifiers discussed in ELENA. In other words, the performance of MLP seems to be a good reference in a benchmark.

In addition, MLP is the most frequently and successfully used network in the neural network community. Therefore, MLP poses an important challenge for any new network or classification algorithm. A work focusing on MLP is the benchmark published by Prechelt [9]. This article comprises classification results for MLP and benchmark datasets of real-world problems (PROBEN1) that can be obtained via FTP. In this article we will compare the results of the MLP on the PROBEN1 datasets with the results of GNG, GCS, and FAM on the same datasets.

## II. Introduction to the Neural Networks Used

For a better comparison of our performance evaluation an introduction to the networks is given. This brief overview mainly gives a description of the training algorithm, the number of parameters and their meaning (see [10] for details).

Manuscript received October 11, 1997; revised July 1, 1998. This work was done at the Department of Neuroinformatik, Technische Universität Ilmenau (Germany).

D. Heinke is with the Cognitive Science Centre, School of Psychology, University of Birmingham, Birmingham B15 2TT, U.K.

F. H. Hamker is with the Department of Neuroinformatics, Technical University of Ilmenau, 98684 Ilmenau, Germany.

Publisher Item Identifier S 1045-9227(98)08812-2.

### *A. Multilayer Perceptrons*

The reference network in this article is the well-known MLP. This network, including the training algorithm called backpropagation, was first introduced by [11]. Since then, a variety of different training algorithms has been developed (see, e.g., [12], [13]).

Prechelt [9] uses the resilient propagation algorithm (RPROP) introduced in [14]. This algorithm performs a local adaptation of the weight-updates according to the behavior of the error function. This is achieved by an individual update-value for each weight. This adaptive update-value evolves during the learning process in such a way that as long as the sign of the partial derivative of the error function stays the same, the adaptive value is increased; otherwise, the adaptive value is decreased. This method assumes that each time the sign changes the last update was too big and the gradient-descent jumped over a local minimum. This algorithm is similar to Quickprop [15], but requires less adjustment of parameters to be stable [14], [9]. It is an epoch learning method and is therefore a good method for medium and small training sets such as those of PROBEN1.

### *B. Fuzzy ARTMAP (FAM)*

Fuzzy ARTMAP [16] is connected with a whole series of neural networks developed on the basis of the adaptive resonance theory (ART) introduced by Grossberg [17]. FAM is capable of supervised learning and consists of two ART networks which form a three-layered network.

The input ART network utilizes fuzzy rules in order to define the similarity between input vectors and weight vectors. The fuzzy rules lead to a partition of the input vector space into hyperrectangles defined by the weight vectors. The maximum size of the hyperrectangles is determined by the vigilance parameter. The output ART network can combine different hyperrectangles in order to connect them with their common class. This is necessary if unconnected regions in the input space belong to the same class.

The learning algorithm for the input ART shows the following behavior: First, it decides if the current input vector is close enough to any of the existing hyperrectangles, as determined by the vigilance parameter. If it is, the closest weight vector is modified so that the corresponding region comprises the input vector as well. If none of the current hyperrectangles is close enough a new region is initialized as the locus of the input vector. The learning algorithm for the output ART compares the class the input vector is mapped onto with the class of this input defined by the data set. If the classes are different, the input ART is forced to introduce a new region and this region is connected with the correct class. If the input ART has already introduced a new hyperrectangle, this region is connected with the correct class.

### *C. Growing Cell Structures (GCS)*

The GCS [18] can perform either unsupervised or supervised learning. The supervised version of GCS combines two families of networks, Kohonen feature map [19] and radial basis functions network [20], with the ability to grow during the training process. It consists of three layers including one hidden layer. The hidden layer forms a topological structure defined, in general, through hypertetrahedrons. This is a special type of neighborhood definition of the Kohonen feature map. Its units have a Gaussian activation function, in which the weights define the center of the Gaussian function and the mean distance between a unit and all of its neighbors defines the activation radius. The output of the network is simply determined by a weighted sum of the activation of the hidden layer. The output unit with the largest activation gives the classification result.

The learning algorithm for the hidden layer has two parts: the first part is a Kohonen-style learning rule, where the best-matching unit and its neighbors are moved toward the input vector. For the best-matching unit and the neighbors there are two different learning rates defined. The second part concerns the insertion and removal of hidden layer units. First, each time a unit is a best-matching unit the “signal counter” of this unit is increased. Second, in each adaption step all signal counters are decreased by a given fraction. Finally, after a fixed number of adaption steps in the space between the unit with the largest signal counter and its most distant neighbor, a unit is inserted. A unit is removed if, roughly speaking, the signal counter falls below a given threshold. The learning algorithm of the output layer is the well-known Delta Rule.

### *D. Growing Neural Gas (GNG)*

The GNG [21] (see [22] for a similar approach) has its origin in the neural gas algorithm [23] and in the GCS. In [10] the original GNG [21] is extended to a supervised network as proposed in [18]. The hidden layer of GNG is also based on a graph but requires no determination of the network dimensionality beforehand as in GCS. Instead, the network starts with two units at random positions and inserts new nodes at positions with high errors. These nodes are connected by edges with a certain age. Old edges are removed, resulting in a topology preserving map with an induced Delaunay triangulation. Similar to the GCS, after a fixed number of steps multiplied with the actual number of nodes a new node is inserted between the location with the highest error or signal counter and the highest one of all its neighbors. The algorithm does not have a deletion criterion. Pattern presentation and calculation of the hidden layer activity are the same as in the GCS. Only the computation of the output activation has changed, slightly, by using a trigonometric function (see [10] for details).

### *E. Discussion*

All networks discussed show a similar three-layer structure with an input layer, a hidden layer, and an output layer. Within this topology the learning method of MLP performs a global adaptation to the training dataset, whereas the incremental networks perform a local adaptation. This might lead to a better generalization property of MLP than with the incremental networks. Hence, MLP might achieve better classification results than GNG, GCS, and FAM. At the same time the learning methods of FAM, GCS, and GNG enable them to insert automatically new nodes in the hidden layers and, consequently, eliminate the crucial parameter of number of hidden layer units.

The incremental networks have a similar learning method. Each of them has an unsupervised learning component in the hidden layer. Here, each weight vector defines a compact area in which the corresponding node produces high output activity. The third layer maps these different areas onto a classification result. However, within this framework FAM, GNG and GCS follow different rules: The unsupervised learning of GCS and GNG are more statistically oriented, whereas FAM performs a more geometrical learning. The similarity measure in FAM is a Fuzzy rule which forms class regions based on hyper-rectangles, whereas GNG and GCS use the Euclidean distance which forms radial-based regions. Comparing GCS and GNG, one major advantage of GNG is its adaptive graph, particularly when deleting nodes.

## III. Structure Analysis of the Benchmark Data

### A. Data Structure Analysis

This section introduces the basic measures on which the analysis of the benchmark data we use is based. The analysis aims at characterizing the dataset in terms of *overlapping* and *complex boundaries* [8]. *Overlapping* indicates how much the data vectors of the different classes interfere with each other. The *complex boundary* indicates to what extent a decision boundary between different classes is simple, e.g., straight lines. These characteristics are evaluated in the following paragraphs with the dispersion and the confusion matrix.

*1) Dispersion Matrix:* The dispersion matrix is a classical measure for overlapping between classes [24]. For computing the dispersion matrix first the within-class inertia of the data set must be computed: Let us assume the database consists of $N$ samples with $c$ classes. The class $\omega_i$ has $N_i$ samples and a center of gravity $\underline{\boldsymbol{g}}_i$ of the input patterns $\underline{\boldsymbol{x}}$

$$I_i = \frac{1}{N_i}\sum_{l=1}^{N_i} |\underline{\boldsymbol{x}}(l) - \underline{\boldsymbol{g}}_i|^2;\ \text{class}(\underline{\boldsymbol{x}}(l)) = \omega_i$$

where $|\cdot|$ is the Euclidean norm.

The dispersion matrix is computed by

$$\text{dispersion}(i,j) = \frac{|\underline{\boldsymbol{g}}_i - \underline{\boldsymbol{g}}_j|}{\sqrt{I_i}.}$$

If the dispersion measure between two classes is large, the classes hardly overlap. If the measure between class $i$ and $j$ is close to one or even lower, class $i$ might have a strong overlap with class $j$. But this is not necessarily the case, if the classes are multimodal and have complex boundaries. Therefore, the measure is necessary but not sufficient for determining the overlap and an additional measure is needed. This additional measure is given by the confusion matrix.

*2) Confusion Matrix:* The confusion matrix is computed by a $k$-nearest neighbor classifier (KNN) [25] together with the error counting method: First, the $k$ is determined by the minimal classification error on the validation dataset. Second, the resulting $k$ is used to compute the confusion matrix of the test dataset. This procedure ensures a certain generalization ability of the $k$-nearest neighbor classifier and characterizes the test dataset. The confusion matrix $\underline{\boldsymbol{C}}$ is

$$C_{ij} = N(\text{class} = j|\text{class} = i) \tag{1}$$

which is the normalized number of the classification result $j$, if the class $i$ is given.

If the confusion value between two classes is close to 100%, the two classes are overlapping. If the confusion value is small and the corresponding dispersion value is small, the overlap indicated by the dispersion matrix is not confirmed by the confusion matrix. This suggests a complex boundary between theses classes.

The $k$-nearest neighbor classifier was chosen for two reasons: First, it is a good, practical reference for the Bayesian error which is needed for computing a good confusion matrix [8]. Second, here, the confusion matrix is used for qualitative characterization of the datasets. This should be done by a classifier that is not the main focus of this paper.

TABLE I
Dispersion Matrix of the Dataset `cancer`

| Class | 1 | 2 |
|---|---|---|
| 1 | 0 | 0.50 |
| 2 | 0.91 | 0 |

### B. The Benchmark Data

We used four types of classification data presented in [9]. These datasets were chosen because they show different degrees of difficulties in terms of overlapping and complexity of boundaries.

For application of the validation method the datasets were divided into three sets: test set, training set, and validation set. The different sets are built from three different partitions of the dataset: training set, 50%, test set and validation set, 25% each. This partition is applied to three different orders of the whole dataset, leading to three different sequences for training, `cancer1`, `cancer2`, and `cancer3` [9]. For each of these sequences the structure is analyzed.

*1) Cancer:* The first classification problem is a diagnosis of breast cancer originally obtained from the University of Wisconsin Hospitals, Madison, from Dr. W. H. Wolberg [26]. The dataset consists of nine inputs, two outputs and 690 examples and is called `cancer`. Examples of input parameters are the clump thickness, the uniformity of cell size and cell shape, the amount of marginal adhesion, and the frequency of bare nuclei. The output classifies the data as either benign or malignant based upon cell descriptions collected by microscopic examination. The dispersion matrix of the whole `cancer` set (Table I) suggests a slight overlap between the two classes, or at least a complex boundary.

Table II shows a reasonably equal distribution of classes and samples. To determine the confusion matrix of `cancer1`, the $k$ of the KNN classifier is set to 12 with an error for the validation dataset of 1.72%. This results in a confusion matrix with a test set error of 1.72%. For `cancer2` $k = 5$

TABLE II
DISTRIBUTION OF SAMPLES IN `cancer`

| Dataset | cancer1 | | cancer2 | | cancer3 | |
|---|---|---|---|---|---|---|
| | 1 | 2 | 1 | 2 | 1 | 2 |
| TrainSet | 229 | 121 | 228 | 122 | 243 | 107 |
| ValidSet | 120 | 55 | 115 | 60 | 107 | 68 |
| TestSet | 109 | 65 | 115 | 59 | 108 | 66 |

TABLE III
CONFUSION MATRIX IN `cancer`

| Dataset | cancer1 | | cancer2 | | cancer3 | |
|---|---|---|---|---|---|---|
| | 1 | 2 | 1 | 2 | 1 | 2 |
| 1 | 100.0 | 0.0 | 95.7 | 4.4 | 98.2 | 1.9 |
| 2 | 4.6 | 95.4 | 3.4 | 96.6 | 9.1 | 90.9 |

TABLE IV
DISPERSION MATRIX OF THE DATASET `diabetes`

| Class | 1 | 2 |
|---|---|---|
| 1 | 0 | 0.37 |
| 2 | 0.58 | 0 |

TABLE V
DISTRIBUTION OF SAMPLES IN `diabetes`

| Dataset | diabetes1 | | diabetes2 | | diabetes3 | |
|---|---|---|---|---|---|---|
| | 1 | 2 | 1 | 2 | 1 | 2 |
| TrainSet | 127 | 257 | 127 | 257 | 129 | 262 |
| ValidSet | 71 | 121 | 71 | 121 | 68 | 124 |
| TestSet | 70 | 122 | 70 | 122 | 78 | 114 |

is selected with an error for the validation dataset of 1.14% and an error of 4.02% on the test set. The $k$ of the KNN classifier for `cancer` is five with a validation dataset error of 3.43%. Based upon the test set, the error is 4.60%. Table III indicates that all `cancer` sets have complex boundaries and only small overlaps. Hence, `cancer` is a well-behaved dataset with complex decision boundaries.

*2) Diabetes:* The second classification problem concerns the diagnosed diabetes of Pima Indians. The `diabetes` dataset has eight inputs, two outputs, and 768 examples. Based upon personal data (age, number of times pregnant) and the results of medical examinations (e.g., blood pressure, body mass index, result of glucose tolerance test, etc.), it states, whether the Pima indian individual is diabetes positive.

This dataset contains some zero elements that seem to replace missing values. As with the `cancer` dataset, the dispersion matrix of the diabetes set (Table IV) suggests a strong overlap between the two classes, or at least a complex boundary.

The distribution of samples and classes of the `diabetes` set is reasonably uniform (Table V). The best KNN ($k = 3$) classifier for `diabetes1` has a validation error of 26.0% and a test error of 25.5%. The confusion matrix confirms the high overlap of class 1 with class 2, where most of the samples of class 1 are expected to belong to class 2; it does not confirm the overlap of class 2 with class 1.

TABLE VI
CONFUSION MATRIX IN `diabetes`

| Dataset | diabetes1 | | diabetes2 | | diabetes3 | |
|---|---|---|---|---|---|---|
| | 1 | 2 | 1 | 2 | 1 | 2 |
| 1 | 44.3 | 55.7 | 30.8 | 69.2 | 60.9 | 39.1 |
| 2 | 8.2 | 91.8 | 11.4 | 88.6 | 14.6 | 85.3 |

TABLE VII
DISPERSION MATRIX OF THE DATASET `glass`

| Class | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | 0 | 0.13 | 3.38 | 5.82 | 8.46 | 1.95 |
| 2 | 0.14 | 0 | 3.54 | 5.97 | 8.75 | 1.96 |
| 3 | 0.77 | 0.78 | 0 | 0.99 | 1.39 | 0.72 |
| 4 | 0.89 | 0.88 | 0.66 | 0 | 0.74 | 0.68 |
| 5 | 0.91 | 0.91 | 0.66 | 0.52 | 0 | 0.76 |
| 6 | 0.77 | 0.74 | 1.23 | 1.74 | 2.79 | 0 |

TABLE VIII
DISTRIBUTION OF SAMPLES AND CLASSES OF `glass1`

| Class | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| TrainSet | 36 | 37 | 6 | 11 | 6 | 11 |
| ValidSet | 20 | 16 | 7 | 2 | 1 | 8 |
| TestSet | 14 | 23 | 4 | 0 | 2 | 10 |

The confusion matrix of `diabetes2` is computed with $k = 9$ with a validation error of 26.04% and a test error of 34.9%. This confirms again an overlap of class 1 with class 2, higher than on `diabetes1` and little overlap between class 2 and class 1 is indicated.

In `diabetes3` the overlap is not as high as in `diabetes1` and `diabetes2`. The confusion matrix is evaluated with $k = 12$, the validation error is 26.56% and the test error is 23.44%.

*3) Glass:* The third dataset gives examples from the classification of glass based upon the description of glass splinters. The `glass` dataset consists of nine inputs, six outputs and 214 samples. It comprises the result of a chemical analysis of glass splinters (percent content of eight different chemical elements) plus the refractive index. These are used to classify the sample to be either float processed, or nonfloat processed relevant knowledge for building windows, vehicle windows, containers, tableware or head lamps. This data set is motivated by forensic needs in criminal investigation.

The dispersion matrix in Table \ref{tab:dispersion_glass} shows a contradictory picture. There exists no overlap between certain classes, e.g. class 5 and class 1. But it also suggests that there are several overlappings, e.g., class 4 / class 3 and class 1 / class 2. The relationship between the number of samples and the size of output and input indicates *a priori* a problem with this dataset. There are not enough samples to achieve a good partition into training, validation and test sets, as illustrated by Table \ref{tab:distribution_glass1}, \ref{tab:distribution_glass2} and \ref{tab:distribution_glass3} for the different partitions of \verb+glass+.

This lead to problems for the KNN in evaluating the confusion matrix. Due to the small dataset and because the

TABLE IX
Confusion Matrix of `glass1`

| Class | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | 78.6 | 21.4 | 0.0 | 0.0 | 0.0 | 0.0 |
| 2 | 30.4 | 69.6 | 0.0 | 0.0 | 0.0 | 0.0 |
| 3 | 25.0 | 75.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 4 | – | – | – | – | – | – |
| 5 | 0.0 | 0.0 | 0.0 | 50.0 | 0.0 | 50.0 |
| 6 | 20.0 | 20.0 | 0.0 | 0.0 | 0.0 | 60.0 |

TABLE X
Distribution of Samples and Classes of `glass2`

| Class | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| TrainSet | 36 | 33 | 8 | 11 | 5 | 14 |
| ValidSet | 15 | 23 | 4 | 0 | 3 | 9 |
| TestSet | 19 | 20 | 5 | 2 | 1 | 7 |

TABLE XI
Confusion Matrix of `glass2`

| Class | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | 73.7 | 26.3 | 0.0 | 0.0 | 0.0 | 0.0 |
| 2 | 40.0 | 55.0 | 0.0 | 5.0 | 0.0 | 0.0 |
| 3 | 100.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 4 | 0.0 | 50.0 | 0.0 | 50.0 | 0.0 | 0.0 |
| 5 | 0.0 | 0.0 | 0.0 | 100.0 | 0.0 | 0.0 |
| 6 | 16.7 | 0.0 | 0.0 | 0.0 | 16.7 | 66.7 |

TABLE XII
Distribution of Samples and Classes of `glass3`

| Class | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Trainset | 29 | 43 | 7 | 9 | 6 | 13 |
| ValidSet | 18 | 21 | 3 | 0 | 3 | 9 |
| TestSet | 23 | 12 | 7 | 4 | 0 | 7 |

KNN is a statistically based classifier, it can not detect the rules underlying the distribution of classes. Thus, this dataset poses a high demand on the ability of a classifier to achieve a generalization. The confusion matrix in Table IX is computed with $k = 23$, a validation error of 29.63% resulting in a test error of 37.74%. Because $k$ is much higher than the number of samples of four classes in the test set, small classes are ignored in favor of often-appearing classes.

On `glass2` the validation dataset is more suitable, supported by the best KNN with $k = 4$. Thus, small classes are not necessarily ignored. Nevertheless, this leads to a high validation error of 42.59% and a test error of 43.40%. The resulting confusion matrix is shown in Table XI. There is a high overlap between class 1 and class 2. Although small classes could have been recognized, there are severe problems in detecting them. Thus, the small classes also have high overlap with other classes.

`Glass3` seems to be less difficult than the previous ones (Table XII). With $k = 3$, a validation error of 33.33% and a test error of 33.96%, the confusion matrix is shown in Table XIII. There is an overlap between the larger classes class 1 and class 2, but less than in the previous sets. Other overlaps exist from class 3 with class 1 and class 2 and also class 4 with class 2.

TABLE XIII
Confusion Matrix of `glass3`

| Class | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | 82.6 | 17.4 | 0.0 | 0.0 | 0.0 | 0.0 |
| 2 | 25.0 | 75.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 3 | 71.4 | 28.6 | 0.0 | 0.0 | 0.0 | 0.0 |
| 4 | 0.0 | 75.0 | 0.0 | 25.0 | 0.0 | 0.0 |
| 5 | – | – | – | – | – | – |
| 6 | 14.3 | 0.0 | 0.0 | 0.0 | 0.0 | 85.7 |

TABLE XIV
Dispersion Matrix of the Dataset `thyroid`

| Class | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 0 | 0.59 | 0.98 |
| 2 | 1.35 | 0 | 0.95 |
| 3 | 41.87 | 17.75 | 0 |

TABLE XV
Distribution of Samples in `thyroid`

| Dataset | thyroid1 | | | thyroid2 | | | thyroid3 | | |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| Trainset | 91 | 181 | 3328 | 88 | 192 | 3320 | 82 | 189 | 3329 |
| ValidSet | 35 | 96 | 1669 | 38 | 86 | 1676 | 42 | 78 | 1680 |
| TestSet | 40 | 91 | 1669 | 40 | 90 | 1670 | 42 | 101 | 1657 |

TABLE XVI
Confusion Matrix of `thyroid1`

| Class | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 50.0 | 15.0 | 35.0 |
| 2 | 4.4 | 12.2 | 83.3 |
| 3 | 0.5 | 0.8 | 98.7 |

*4) Thyroid:* The fourth and last dataset is called `thyroid` and has 21 inputs, three outputs, and 7200 examples. In contrast to all other sets the `thyroid` set contains nine Boolean input values and four other inputs which are often exactly zero, and some others which are nonnormalized values. The problem underlying this dataset is to diagnose thyroid hyper- or hypofunction based upon patient query data and patient examination data.

The dispersion matrix of all samples is shown in Table XIV. Class 1 shows possible overlap between class 2 and class 3. Class 2 has some overlap with class 3 and less with class 1. This dataset includes many more samples of class 3 than the others.

The distribution of the samples shows a similar problem to that with `glass`. Class 3 is so dominant that class 1 and class 2 might be ignored by a statistically based classifier. In each of the partial sets `thyroid2` and `thyroid3` contain nearly the same number of samples in each class as does `thyroid1`.

Because of the large size of the `thyroid` data sets, only four $k$'s for the KNN classifier were tested on the `thyroid1` set. This test shows that $k = 3$ results in a minimal validation error of 5.67% and a test error of 6.67%.

The confusion matrix (Table XVI) confirms the overlap between class 1 and class 3 and a small overlap between class1 and class 2 as indicated by the dispersion matrix, but does not confirm the overlap between class 1 and class 2. `thyroid2` and `thyroid3` show the same kind of overlap.

*5) Summary:* The four datasets represent different levels of difficulty of classification. `Cancer` is a relatively easy classification problem, with complex boundaries between classes, only little overlap between classes, and sufficient number of data points. `Diabetes` increases the degree of difficulty by consisting of overlapping classes in addition to complex boundaries. `Glass`, in addition to complex boundaries and overlapping of classes, shows a lack of sufficient data. The same can be stated for `thyroid`. However, `thyroid` shows the additional feature of having linear boundaries between the classes due to Boolean input variables. The linearity can be considered a facilitation of classification, but as the following sections show poses, it poses some difficulties to the GNG and GCS.

## IV. Benchmarking Rules

For a better comparison and evaluation of this benchmark all equations and benchmarking procedures are introduced.

### A. General Benchmark Rules

The main comparison between the networks is based upon their classification error. Each network was trained with different initial values, different parameters and different order of the training dataset. The details on the training methods for each network are given in the following sections. After the training the mean of classification error on the test dataset $(\overline{e})$ and its standard deviation $(\sigma)$ on validation and test dataset were determined. The corresponding header of the tables is called *mean test* $(\sigma)$. In addition, the best run on the test dataset is computed and called *test best*.

The statistical comparison of the network behavior is performed with a *t-test* [27]. The $t$-test is a parametric test that compares the mean values of two samples. The test is appropriate even when the variance is unknown and the samples are small and independent. However, it assumes that the distribution of the underlying population is normal. The extension of the $t$-test used here is also suitable when the variances of the samples are different, which is true in most cases. The application of the $t$-test answers the question, whether network $A$ performs significantly better than network $B$ on average. Because these classification errors are usually approximately log-normal distributed, the classification errors of the networks are logarithmically transformed. This transformation tries to meet the normal distribution condition and because it is strictly monotone the transformation does not change the test result. Finally, the $t$-test is performed with the hypothesis: logarithmic error of network $A$ and $B$ are the same, versus logarithmic error of network $B$ is larger than error of network $A$. If the resulting $P$-value of the $t$-test indicates a significant result $(P < \alpha)$, network $A$ performs significantly better than network $B$ on average. If the result is not significant, a similar performance of network $A$ and $B$ can be ruled out. In analysis of the networks, performance the significance level $\alpha$ was set to 0.1.

Apart from the classification error three additional performance measures were used: the mean number of inserted nodes, the mean number of epochs used for training and the effect of variation of the parameters. The first performance measure is only appropriate for the incremental networks as it evaluates their insert and removal criteria. In addition, a lower number of nodes means fast computation of classification results and less memory requirement, which might be important in applications. The second performance measure gives an impression of how efficient the learning algorithm performs. The third performance measure is an important measure, since finding the “correct” parameters for the network determines the success or failure of the training process. We use the standard deviation of the classification error on the test dataset as a measure for the effect of parameter variations on the behavior of the network.

### B. Multilayer Perceptron

*1) Selected Results:* In [9] a “two-step benchmark test” was done: First, 12 different kinds of network topologies were used including one hidden layered, two hidden, layered and short cut architectures. According to [9] the number of runs for these architectures were too few in order to decide which architecture is the significant best. Therefore, the largest architecture of the 5%-best architectures was chosen and 60 runs were performed with these so-called pivot architectures. In both steps two different measures were published. In the first step only the best classification errors were documented and in the second benchmark step the mean and the standard deviation were listed.

In order to perform a complete comparison with our results both results in [9] were included into the comparison. Hence, the comparison with MLP made here can be understood qualitatively only and the order of magnitude of the values is meaningful.

*2) Validation Method:* For controlling the training process the early stopping method was used [9]: Training was stopped when the stopping criterion $GL_\alpha$ was fulfilled or when a maximum of 3000 epochs occurred. The stopping criterion $GL_\alpha$ is fulfilled when $GL$ exceeds the threshold $\alpha$

$$GL(t) = 100\left(\frac{E_{va}(t)}{E_{\text{opt}}(t)} - 1\right). \tag{2}$$

Exceeding the threshold might indicate that a loss of generalization ability has occurred. $\alpha$ was set to five.

The minimum validation error $E_{\text{opt}}(t)$ is obtained by

$$E_{\text{opt}}(t) = \min_{t' \leq t} E_{va}(t'). \tag{3}$$

The current validation error is squared error percentage (SEP)

$$E = 100 \cdot \frac{\zeta_{\max} - \zeta_{\min}}{n \cdot p} \sum_{j=1}^{p} \sum_{i=1}^{n} (o_{ji} - \zeta_{ji})^2. \tag{4}$$

Here $p$ denotes the number of patterns.

TABLE XVII
FREE PARAMETERS OF GCS FOR glass, cancer, AND diabetes DATA

| Param Set | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| network dimension | 2 | 2 | 2 | 3 | 3 | 3 |
| learning rate of best | 0.1 | 0.1 | 0.2 | 0.1 | 0.1 | 0.2 |
| learning rate of neighbours | 0.006 | 0.006 | 0.012 | 0.006 | 0.006 | 0.012 |
| learning rate of output | 0.15 | 0.15 | 0.15 | 0.1 | 0.1 | 0.1 |
| adaption steps | 100 | 200 | 200 | 100 | 200 | 100 |
| decreasing counters | 0.995 | 0.995 | 0.995 | 0.995 | 0.995 | 0.995 |
| removal threshold | 0.005 | 0.005 | 0.005 | 0.005 | 0.005 | 0.005 |

TABLE XVIII
FREE PARAMETERS OF GNG FOR ALL DATA SETS

| Param Set | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| learning rate of best | 0.1 | 0.1 | 0.2 | 0.1 | 0.2 | 0.1 |
| learning rate of neighbours | 0.006 | 0.006 | 0.012 | 0.006 | 0.012 | 0.006 |
| learning rate of output | 0.15 | 0.15 | 0.15 | 0.3 | 0.3 | 0.15 |
| adaption steps | 100 | 200 | 200 | 100 | 100 | 10 |
| decreasing counters | 0.995 | 0.995 | 0.995 | 0.995 | 0.995 | 0.995 |
| decreasing of signal counters | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
| maximal age of the edges | 50 | 50 | 50 | 50 | 50 | 50 |

### *C. Fuzzy ARTMAP (FAM)*

*1) Simulation Parameters:* The value of the vigilance parameter $\rho$ was restricted to be less than or equal to 0.6, because the size of the hyperrectangles becomes too small with larger vigilance values. When this happens, each hyperrectangle represents little more than on data point, and the ability of generalization is reduced. In order to decrease the possible variations of parameters the values of the vigilance parameter were set to 0, 0.3, 0.5, or 0.6.

*2) Validation Method:* Preliminary simulations showed that FAM always converged to a stable state and that this happened fast. Because early stopping can not be applied in this case, the following control of the training process was used: FAM was trained with a certain parameter set until it converged to its stable state (number of epochs). After that, the status with the lowest validation set error was chosen and the corresponding test set error was computed. Here 40 runs were performed with ten different orders of the training dataset, with ten runs for each of four vigilance parameter values.

### *D. Growing Cell Structures*

*1) Simulation Parameters:* Preliminary runs with several variations of parameters have revealed a robustness against slightly changing values, therefore, not all parameters were varied in different runs. Others, as seemed appropriate, were set to fixed values in all simulations. Each network was trained five times with six different parameter sets (Table XVII). The initialization and the order of presentation were random. This means that for each data set, 30 runs were performed. The maximum number of epochs in GCS simulations was 50, because preliminary simulations showed that longer training does not lead to a better performance.

For the thyroid dataset slightly different parameters were used. Because of instable output weights in simulations with the thyroid data set, the learning rate was set to 0.01. Because of the huge amount of training data in thyroid the maximum number of epochs was decreased to 20. All other parameters remained the same.

*2) Validation Method:* First, the early stopping method used with MLP was tested. But this led to a premature termination of the training process and therefore the early stopping method had to be modified. Each network was trained up to a given maximum of epochs. During the training process, after every five epochs the $GL_\alpha$ stopping criterion was applied. From the networks that fulfilled $GL_\alpha$ the one with the minimum validation squared error percentage (SEP) was chosen to be the best of that run; $\alpha$ was set to five.

### *E. Growing Neural Gas*

*1) Simulation parameters:* Similarly to the GCS, the GNG consists of several free parameters. Because preliminary runs have shown that only some parameters have a strong influence on the outcome of the training, only a few parameters were varied (see Table XVIII). Again, for each dataset 30 runs were performed: Six runs of every parameter set, each with five different randomly chosen initializations. The maximum number of epochs in GNG simulations was 200.

*2) Validation Method:* The same validation method as with GCS was used.

## V. RESULTS

In this chapter the results are discussed under the aspects of different performance measures. As introduced in the benchmark rules, the performance measures are: classification error on the test datasets, the effect of variations of parameters on the behavior of the networks, the number of training epochs and the number of inserted nodes. Table XIX summarizes the

TABLE XIX
This Table States the Best Network for Every Performance Measure on Each Dataset. The First Three Columns Show the Result on the Classification Error (Best Classification Error, Best Mean of Classification Error and Significance of Mean). The Third Column States the Network with the Fewest Training Epochs. The Fourth Column Notes the Network with Fewest Inserted Nodes. The Last Column Gives the Network with the Smallest Standard Deviation Which Is Used as Measure for the Robustness of the Networks Against Parameter Variations. Please Note, That MLP is Neither Included in the Significance Test Because the Necessary Details Are not Available nor in the Comparison of Nodes Because of Its Incompatibility with the Incremental Networks

| Dataset | best-min | best-mean | signif.-worse | best-epoch | best nodes | best $\sigma$ |
|---|---|---|---|---|---|---|
| cancer1 | GCS | MLP | FAM | FAM | FAM | MLP |
| cancer2 | FAM | GCS | – | FAM | FAM | GNG |
| cancer3 | MLP | GCS | FAM | FAM | FAM | GNG |
| diabetes1 | GCS | MLP | FAM | FAM | FAM | GNG |
| diabetes2 | MLP | MLP | – | FAM | FAM | GNG |
| diabetes3 | MLP | MLP | FAM | FAM | FAM | GNG |
| thyroid1 | FAM | MLP | GNG, GCS | FAM | FAM | GCS |
| thyroid2 | MLP | MLP | GNG, GCS | FAM | FAM | GNG |
| thyroid3 | MLP | MLP | GNG, GCS | FAM | GNG | MLP |
| glass1 | GCS | GNG | – | FAM | FAM | FAM |
| glass2 | FAM | GNG | – | FAM | FAM | MLP |
| glass3 | GCS | GCS | – | FAM | FAM | FAM |

TABLE XX
Comparison of Networks on the `cancer` Datasets. Best Result on the Test Set; Mean Result and stdv. on the Test Set; Mean Number of Epochs Trained; the Mean Number of Inserted Nodes

| Network | dataset | test-best | test-mean($\sigma$) | epoch-mean($\sigma$) | nodes |
|---|---|---|---|---|---|
| GCS | 1 | 0.60 | 1.84 (0.98) | 41.00 (8.75) | 105.80 |
| GNG | 1 | 1.15 | 1.70 (0.39) | 163.17 (27.62) | 43.07 |
| FAM | 1 | 1.72 | 6.40 (2.73) | 3.45 (1.28) | 23.12 |
| MLP | 1 | 1.15 | 1.47 (0.60) | 152 (111) | |
| GCS | 2 | 2.90 | 4.04 (0.69) | 42.17 (10.80) | 110.93 |
| GNG | 2 | 2.90 | 4.22 (0.55) | 159.33 (38.90) | 40.23 |
| FAM | 2 | 1.72 | 6.08 (2.40) | 3.44 (1.16) | 19.86 |
| MLP | 2 | 5.75 | 4.52 (0.70) | 81.00 (72.00) | |
| GCS | 3 | 2.30 | 3.12 (0.64) | 41.33 (8.30) | 107.70 |
| GNG | 3 | 2.30 | 3.13 (0.47) | 132.67 (54.67) | 37.00 |
| FAM | 3 | 2.30 | 7.08 (3.63) | 2.67 (0.76) | 17.77 |
| MLP | 3 | 2.30 | 3.37 (0.71) | 51.00 (16.00) | |

results of this paper. Tables XX–XXIII show the details of the results.

### *A. Classification Error*

In addition to Tables XX–XXIII, Figs. 1–4 give a graphical illustration of the performance on the datasets.

As seen in Table XX and Fig. 1, the best classification error of all networks on `cancer` are in the same range. The same is true for the mean values of the classification error on the test dataset, excluding the result of FAM. For `cancer1` and `cancer3` the $t$-test confirms that FAM performs significantly worse than all other networks (Table XIX). Apparently, on dataset with complex boundaries and no overlap such as `cancer`, GNG, GCS, and MLP performed similarly well, whereas FAM encountered problems.

On the diabetes dataset (Fig. 2, Table XXI) MLP performs slightly better than the other networks, especially on `diabetes2`. Again, the FAM performs the worst; this is shown by the $t$-test to be significant for `diabetes1` and `diabetes3`. However excluding `diabetes3`, FAM with its best network is in the range of the best of GCS and GNG. In contrast to `cancer` the confusion matrix of the `diabetes` sets, especially on `diabetes1` and `diabetes2` indicates high overlap. In particular, `diabetes2` shows the strongest overlap. This matches the slightly worst performance of all networks on `diabetes2`.

The `thyroid` dataset results in a different performance of the networks compared to other datasets (Fig. 3, Table XXII). Here, the GNG and GCS perform significantly worse than MLP and FAM. Interestingly, the results of the FAM and MLP are similar, especially the standard deviation of FAM, which contrary results with all other datasets is comparable to that of the MLP. There are two reasons for the good performance of FAM: First, the number of data samples for certain classes is too small compared to the complexity of the dataset. Statistically oriented classifiers, such as GNG and GCS, tend to have problems with such a dataset, because they might ignore these classes. Geometric methods, such as

TABLE XXI
COMPARISON OF NETWORKS ON THE `diabetes` DATASETS. BEST RESULT ON THE TEST SET; MEAN RESULT AND stdv. ON THE TEST SET; MEAN NUMBER OF EPOCHS TRAINED; THE MEAN NUMBER OF INSERTED NODES

| Network | dataset | test-best | test-mean($\sigma$) | epoch-mean($\sigma$) | nodes |
|---|---|---|---|---|---|
| GCS | 1 | 21.90 | 26.08 (2.72) | 34.83 (9.86) | 89.45 |
| GNG | 1 | 23.40 | 26.01 (1.75) | 153.33 (42.35) | 39.60 |
| FAM | 1 | 23.44 | 33.11 (3.78) | 6.60 (1.06) | 25.95 |
| MLP | 1 | 25.00 | 24.57 (3.53) | 192 (98) | |
| GCS | 2 | 25.52 | 31.61 (3.77) | 32.33 (13.11) | 83.00 |
| GNG | 2 | 26.60 | 30.19 (1.61) | 122.17 (59.78) | 32.00 |
| FAM | 2 | 27.08 | 34.84 (3.39) | 6.65 (1.41) | 26.68 |
| MLP | 2 | 23.44 | 25.91 (2.50) | 119 (42) | |
| GCS | 3 | 22.40 | 26.23 (2.62) | 39.33 (11.94) | 103.43 |
| GNG | 3 | 21.90 | 25.43 (1.85) | 147.67 (49.14) | 43.53 |
| FAM | 3 | 28.12 | 34.23 (3.29) | 6.60 (1.15) | 26.55 |
| MLP | 3 | 21.35 | 23.06 (1.91) | 307 (193) | |

TABLE XXII
COMPARISON OF NETWORKS ON THE `thyriod` DATASETS. BEST RESULT ON THE TEST SET; MEAN RESULT, AND stdv. ON THE TEST SET; MEAN NUMBER OF EPOCHS TRAINED; THE MEAN NUMBER OF INSERTED NODES

| Network | dataset | test-best | test-mean($\sigma$) | epoch-mean($\sigma$) | nodes |
|---|---|---|---|---|---|
| GCS | 1 | 6.30 | 6.79 (0.28) | 20.00 (0.00) | 477.53 |
| GNG | 1 | 6.20 | 7.20 (0.40) | 167.33 (52.34) | 92.03 |
| FAM | 1 | 1.67 | 2.34 (0.52) | 2.33 (0.48) | 32.76 |
| MLP | 1 | 2.00 | 2.32 (0.67) | 491 (319) | |
| GCS | 2 | 6.60 | 6.92 (0.30) | 20.00 (0.00) | 475.86 |
| GNG | 2 | 6.40 | 7.25 (0.21) | 172.41 (45.49) | 93.72 |
| FAM | 2 | 1.28 | 2.20 (0.47) | 2.35 (0.59) | 34.30 |
| MLP | 2 | 1.28 | 1.86 (0.41) | 660 (460) | |
| GCS | 3 | 6.90 | 7.35 (0.22) | 19.83 (0.91) | 474.77 |
| GNG | 3 | 6.70 | 7.74 (0.40) | 185.50 (32.01) | 96.90 |
| FAM | 3 | 1.89 | 2.43 (0.34) | 2.55 (0.51) | 39.60 |
| MLP | 3 | 1.50 | 2.09 (0.31) | 598 (624) | |

TABLE XXIII
COMPARISON OF GCS, GNG, FAM, AND MLP ON THE `glass` DATASETS. BEST RESULT ON THE TEST SET; MEAN RESULT AND stdv. ON THE TEST SET; MEAN NUMBER OF EPOCHS TRAINED; MEAN NUMBER OF INSERTED NODES

| Network | dataset | test-best | test-mean($\sigma$) | epoch-mean($\sigma$) | nodes |
|---|---|---|---|---|---|
| GCS | 1 | 22.60 | 34.09 (6.03) | 43.17 (9.51) | 37.73 |
| GNG | 1 | 22.60 | 33.84 (7.46) | 137.00 (44.07) | 21.90 |
| FAM | 1 | 33.96 | 39.76 (3.59) | 2.20 (0.61) | 21.65 |
| MLP | 1 | 32.08 | 39.03 (8.14) | 67 (44) | |
| GCS | 2 | 32.10 | 43.97 (4.61) | 38.00 (12.36) | 32.63 |
| GNG | 2 | 32.10 | 39.18 (3.89) | 157.33 (36.14) | 21.70 |
| FAM | 2 | 28.90 | 40.82 (8.50) | 2.33 (0.47) | 20.30 |
| MLP | 2 | 52.83 | 55.60 (2.83) | 29 (9) | |
| GCS | 3 | 24.50 | 35.41 (8.30) | 41.17 (10.14) | 35.63 |
| GNG | 3 | 26.40 | 40.30 (9.18) | 162.67 (41.41) | 21.73 |
| FAM | 3 | 30.19 | 44.10 (7.69) | 2.20 (0.61) | 22.30 |
| MLP | 3 | 33.96 | 59.25 (7.83) | 66 (46) | |

FAM, might have no problems with such datasets. This is also supported by the results with `glass` (see below). Second, nine inputs of `thyroid` are boolean variables. Therefore, the boundaries between the different classes can be straight lines. This is more suitable for the hyperrectangle regions generated by the hidden layer of FAM than to radial-based regions of GNG and GCS. The same can be stated for MLP. Apparently, FAM can show good performance if the dataset consists of linear boundaries, or is generated by Boolean input values, contains some classes with a small amount of data.

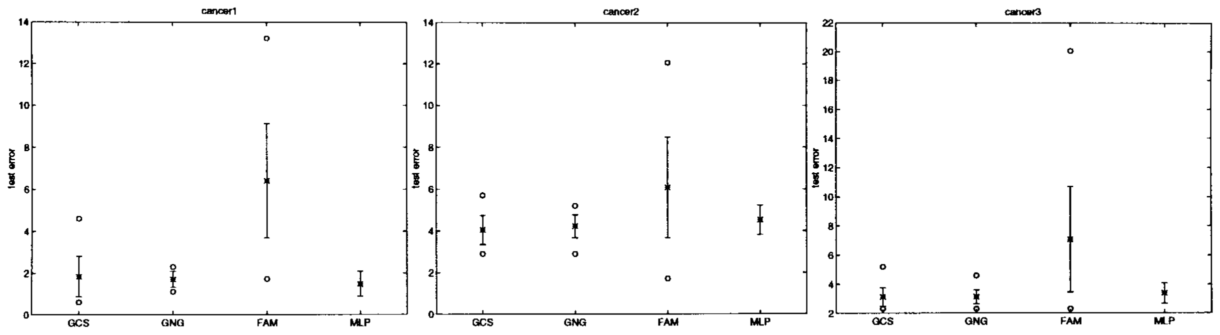


Fig. 1. Mean test error, stdv., best and worst net of the various algorithms (for MLP no best and worst is plotted) on `cancer`.

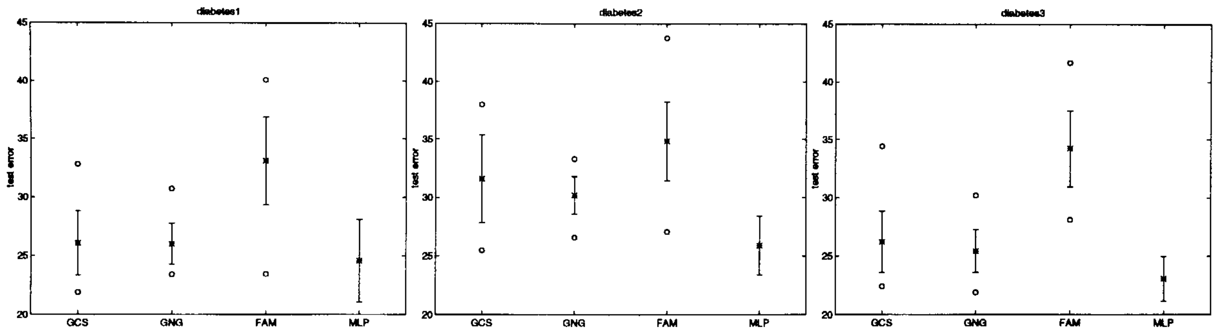


Fig. 2. Mean test error, stdv., best, and worst net of the various algorithms (for MLP no best and worst is plotted) on *diabetes*.

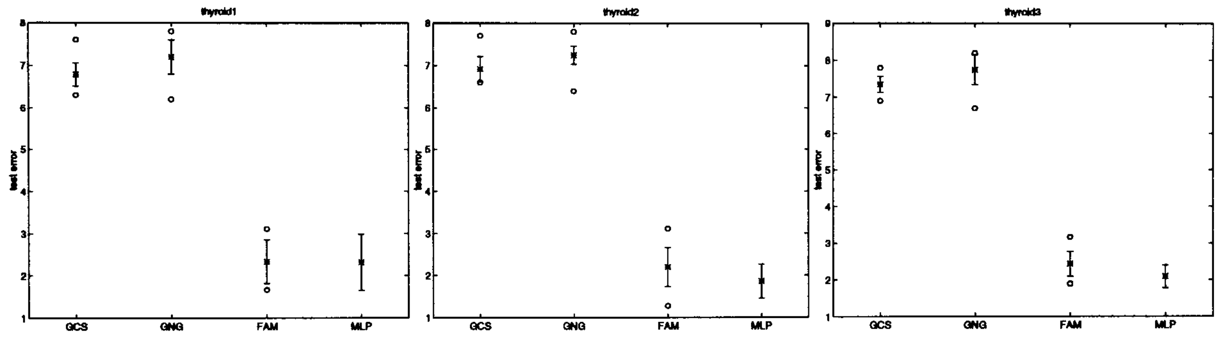


Fig. 3. Mean test error, stdv., best, and worst net of the various algorithms (for MLP no best and worst is plotted) on `thyroid`.

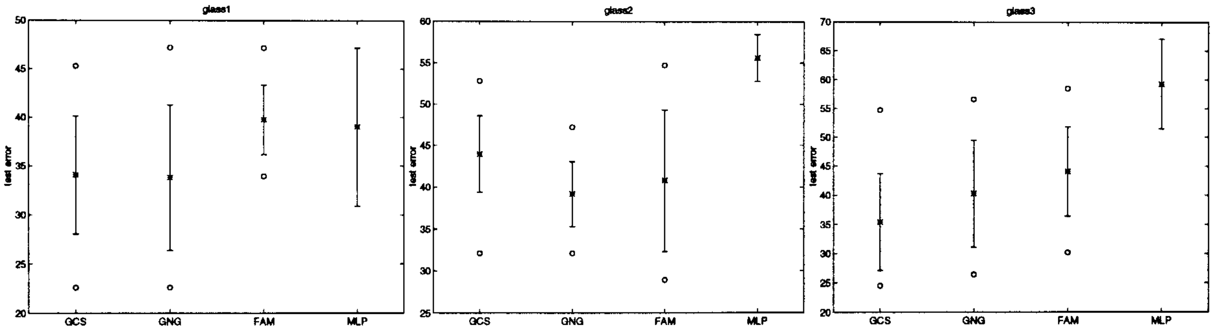
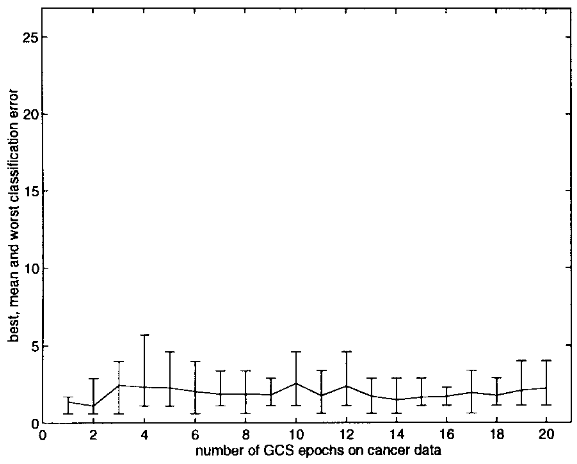


Fig. 4. Mean test error, stdv., best, and worst net of the various algorithms (for MLP no best and worst is plotted) on `glass`.

The `glass` dataset is the only one with which MLP has problems (Fig. 4, Table XXIII). On `glass2` the worst GNG is better than the best MLP, on `glass2` and `glass3` the mean MLP has more errors than the worst GNG, GCS or FAM. Only on `glass1` is the MLP not as bad. GNG, GCS and FAM show no significant difference in result. The high errors of MLP may be due to the small number of samples in this dataset. In a similar line of argument to that applied to `thyroid`, the



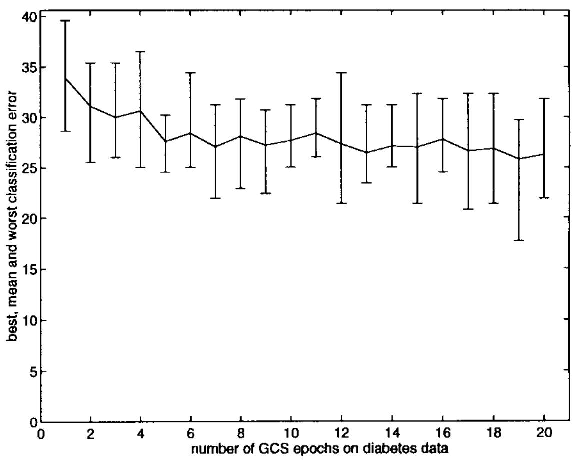


Fig. 5. Mean, best, and worst test error of the GCS in ten runs on the first set of each benchmark dataset with a fixed parameter set and a random initialization of the weights. No validation criteria are used.

geometrically oriented FAM has fewer problems with small sample size than the statistically oriented GNG and GCS. This apparently allows FAM to overcome its generally bad performance. However, in `glass` the shapes of the class boundaries are not as suitable for FAM as in `thyorid`, and therefore FAM shows results similar to those of GNG and GCS.

## B. Number of Inserted Nodes

FAM has the lowest number of nodes in all cases apart from `thyroid3` (Table XIX). However, the number of inserted nodes in GCS and GNG roughly depends on the ratio between trained epochs and adaption steps, since GCS and GNG insert new nodes every epoch. Additional examinations in [10] reveal that GCS and GNG still perform better than FAM with fewer numbers of training epochs, thus, a smaller amount of nodes. Hence, if number of nodes is an important issue, GNG and GCS can be tuned to have fewer nodes without losing much performance.

## C. Variation of Parameters

Table XIX shows that GNG has for most datasets the lowest standard deviation. This is considered to be low sensitivity to variation of parameters. The most prominent exception is `glass`, where FAM is generally better. This might be due to the difference between geometrically oriented and statistically oriented classifiers as discussed above.

In [10] the different effects of the parameters in the different networks are examined further. For FAM, this reveals that the order of the training data is most important for the outcome of the training. This is not surprising, because the order influences the growth of the hyperrectangles which determines the training result. The vigilance parameter plays an important role only when it meets the correct discretization of the dataset. For GNG and GCS the crucial parameters is the adaption step. If the adaption step is chosen too high, no insertions for rare classes tend to occur.

## D. Number of Epochs

For all datasets FAM shows the lowest number of epochs to reach automatic termination of learning. However, detailed analysis of GCS and GNG shows that these networks adapt quickly to the training set. Fig. 5 gives examples for GCS, for details see [10]. Only a few epochs are needed to reach the range of the best results. Further training only slightly improves the result, but a higher amount of hidden nodes emerges. In general, if a network is needed that is not the very best but is fast (while small), a GCS should be used and trained only a few epochs. This is because GCS does not stop automatically as does FAM.

## VI. Conclusion

The present paper began with the question of which is the "best" neural network for solving a pattern classification task—the well-known MLP, or one of the more recently developed incremental networks, FAM, GCS, or GNG? This question was examined in the framework of four real-world datasets representing different levels of difficulty of classification problems. The first dataset (`cancer`) is a relatively easy classification problem with complex boundaries between classes, only little overlap between classes and a sufficient number of data points. The second dataset (`diabetes`) increases the degree of difficulty by having overlapping classes in addition to complex boundaries. The third dataset (`glass`) shows, besides complex boundaries and overlapping of classes, a lack of sufficient data. The same is true for the fourth dataset (`thyroid`). However, `thyroid` shows an additional feature of linear boundaries between the classes due to Boolean input variables.

The reference in this benchmark was the result of the extensive study of MLP by Prechelt [9]. From a theoretical viewpoint, one could expect a better performance for MLP than of the incremental networks because MLP performs a global adaptation to the training dataset, whereas the incremental networks perform a local adaptation. The results of this benchmark show that this is clearly not the case. On the contrary, MLP performs in the same range as the incremental networks. Thus, the elimination of the parameter of the number of hidden nodes through the incremental mechanism outweighs the disadvantage of local adaption in the incremental networks.

Originally, we aimed at finding a clear answer to the question which is the "best" network in terms of the classification error. Since none of the networks consistently performed significantly better than the other networks, there is no clear answer to our question. However, we found some rules which state how well a certain network performs, given certain properties of a dataset. These rules are summarized here.

Except for the fourth dataset MLP, GCS, GNG perform similarly, whereas FAM behaves worse and for the first two datasets this behavior is significantly worse according to the $t$-test. Hence, FAM tends not only to have problems with datasets having overlapping classes but also with datasets having complex boundaries. For the third dataset, despite its overlapping properties, the performance of FAM is not significantly worse, because its more geometrically oriented behavior has fewer problems with the few data points in this dataset than the statistically oriented GNG and GCS.

For the fourth dataset a different picture emerges: GNG and GCS behave significantly worse than MLP and FAM. This is mainly due to the linear boundaries between classes following the Boolean input variables. For these boundaries, the hyperrectangle-based regions of FAM and the polygen-based regions of MLP are more suitable than the radial-based regions of GNG and GCS.

Apart from the classification error, other performance measures were examined in this paper. For the number of inserted nodes, an important performance measure for the incremental networks, FAM performs best. However, the training of GNG and GCS can be tuned so that they insert less units but still perform better than FAM. For the number of epochs FAM shows the shortest training time. However, GNG and GCS also show rapid convergence, whereas MLP typically shows slow convergence. Finally, the dependence of the variation of performance upon variation of parameters was examined. Here, GNG clearly outperforms the other networks. For MLP, the time consuming search of a good architecture and the best choice of parameters, as demonstrated by [9], plays a crucial role. Only for the datasets with few data samples FAM does show fewer variations in behavior than GNG.

In summary, considering the similar classification performance of MLP, GNG, GCS, the rapid convergence of GNG and GCS and the small dependence on variation of parameters of GNG, the overall ranking of networks in descending order is: GNG, GCS, MLP, and FAM. However, when the dataset shows linear boundaries between classes, FAM and MLP can perform better than GNG and GCS.

## Acknowledgment

The authors thank Professor H.-M. Gross and D. Surmeli for comments on a preliminary version. They also wish to thank T. Vesper and S. Wohlfahrt for implementing the algorithms.

## References

[1] A. Flexer, "Statistical evaluation of neural network experiments: Minimum requirements and current practice," in *Proc. 13th European Meet. Cybern. Syst. Res.*, 1996.

[2] F. H. Hamker and H.-M. Gross, "A lifelong learning approach for incremental neural networks," in *14th European Meet. Cybern. Syst. Res. (EMCSR'98)*, Vienna, Austria, 1998.

[3] ——, "Task-relevant relaxation network for visuo-motory systems," in *Proc. ICPR'96—Int. Conf. Pattern Recognition*, Vienna, Austria, 1996, pp. 406–410.

[4] F. Hamker, T. Pomierski, H.-M. Gross, and K. Debes, "Ein visuo-motorisches System zur Loesung der MIKADO-Sortieraufgabe," in *Proc. SOAVE'97—Selbstorganization von adaptivem Verhalten,* Ilmenau, Germany, 1997.

[5] L. Prechelt, "A quantitative study of experimental evaluations of neural network learning algorithms: Current research practice," *Neural Networks*, vol. 9, pp. 457–462, 1996.

[6] B. D. Ripley, "Flexible nonlinear approaches to classification," in *From Statistics to Neural Networks: Theory and Pattern Recognition Applications*. Berlin, Germany: Springer-Verlag, 1993, pp. 105–126.

[7] C. E. Rasmussen, R. M. Neal, G. E. Hinton, D. van Camp, and M. Revow, "The DELIVE Manual Version1.0," Univ. Toronto, Toronto, Ontario, Canada, Tech. Rep. Available: http://www.cs.utoronto.ca/\delve/, 1996.

[8] A. Guérin-Dugué *et al.*, "Deliverable R3-B4-P-Task B4: Benchmarks," Elena-NervesII *Enhanced Learning for Evolutive Neural Architecture*, Tech. Rep., June 1995. Available FTP: ftp.dice.ucl.ac.be/pub/neural-nets/ELENA/Benchmarks.ps.Z

[9] L. Prechelt, "PROBEN1—A set of benchmarks and benchmarking rules for neural network training algorithms," Tech. Report 21/94, Fakultät für Informatik, Universität Karlsruhe, Germany, 1994. Available FTP: ftp.ira.uka.de /pub/papers/techreports/1994/1994-21.ps.Z

[10] F. Hamker and D. Heinke, "Implementation and comparison of growing neural gas, growing cell structures and fuzzy artmap," Tech. Rep., Schriftenreihe des FG Neuroinformatik der TU Ilmenau, Report 1/97, 1997.

[11] D. E. Rumelhart, J. L. McClelland, and PDP Research Group, *Parallel Distributed Processing; Explorations in the Microstructure of Cognition: Volume 1—Foundations*. Cambridge, MA: MIT Press, 1988.

[12] P. J. Werbos, "Backpropagation: Past and future," in *Proc. ICNN-88*, New York, 1988, pp. 343–353.

[13] T. Tollenaere, "Supersab: Fast adaptive backpropagation with good scaling properties," *Neural Networks*, 1990, pp. 561–573.

[14] M. Riedmiller and H. A. Braun, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm," in *IEEE ICNN-93*, San Francisco, CA, 1993, pp. 586–591.

[15] S. E. Fahlman, "Faster learning on backpropagation: An empirical study," in *Proc. 1988 Connectionist Summer School*, 1988, pp. 38–59.

[16] G. A. Carpenter, S. Grossberg, M. Markuzon, J. H. Reynolds, and D. B. Rosen, "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps," *IEEE Trans. Neural Networks*, vol. 3, pp. 698–713, 1992.

[17] S. Grossberg, "Adaptive pattern classification and universal recoding: I. Parallel developement and coding of neural feature detectors," *Biol. Cybern.*, vol. 23, pp. 121–134, 1976.

[18] B. Fritzke, "Growing cell structures—A self-organizing network for unsupervised and supervised learning," *Neural Networks*, vol. 7, no. 9, 1994.

[19] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biol. Cybern.*, vol. 43, pp. 59–69, 1982.

[20] J. Moody and C. Darken, "Learning with localized receptive fields," in *Proc. 1988 Connectionist Models Summer School*, D. Touretzky, G. Hinton, and T. Sejnowski, Eds. San Mateo, CA: Morgan Kaufmann, 1988, pp. 133–143.

[21] B. Fritzke, "A growing neural gas network learns topologies," in *Advances in Neural Inform. Processing Syst.*, G. Tesauro, D. S. Touretzky, and T. K. Keen, Eds. Cambridge, MA: MIT Press, 1995, pp. 625–632.

[22] J. Bruske and G. Sommer, "Dynamic cell structure learns perfectly topology perserving map," *Neural Comput.*, vol. 7, pp. 845–865, 1995.

[23] T. Martinetz and K. J. Schulten, "A neural-gas network learns topologies," in *Artificial Neural Networks*, T. Kohonen, K. Mäkisara, and O. Simula, Eds., Amsterdam, 1991, pp. 397–402.

[24] C. W. Therrien, *Decision Estimation and Classification*. New York: Wiley, 1989.

[25] R. O. Duda and P. E. Hart, *Pattern Classifiaction and Scene Analysis*. New York: Wiley, 1973.

[26] W. H. Wolberg, "Cancer dataset obtained from Wiliams H. Wolberg from the University of Wisconsin Hospitals, Madison."

[27] W. W. Hines and D. C. Montgomery, *Probability and Statistics in Engineering and Management Science*. New York: Wiley, 1990.

**Dietmar Heinke** received the diploma degree in electrical engineering in 1990 from the Technical University Darmstadt, Germany. In 1993 he joined the Department of Neuroinformatics at the University of Ilmenau, Germany, where he received the Ph.D. degree in 1997.

From 1991 to 1992 he work at the Research Institute for Applied Artificial Intelligence (FAW) in Ulm. While in Ilmenau, he visited the Cognitive Science Centre, School of Psychology, University of Birmingham, U.K. as a Research Fellow in 1996. Since 1998 he has been a Postdoctoral Research Fellow at the same institution. His research interests include cognitive neuroscience, computational modeling, visual object recognition, visual attention, and neural networks.

**Fred H. Hamker** received the diploma degree in electrical engineering from the Universität Paderborn, Germany, in 1994. Since then he has worked toward the Ph.D. degree at the Department of Neuroinformatics, Technische Universität Ilmenau (Germany), which is currently under review. The main topics in his thesis cover biological models of attention and neural networks for life-long learning encountering the stability-plasticity-dilemma.

In 1997 he participated at the Workshop on Neuromorphic Engineering in Telluride (USA). Recently he joined the medical data analysis project at the Workgroup of Adaptive System Architecture, Department of Computer Science, J. W. Goethe Universität, Frankfurt am Main.