SOAVE'97 - Selbstorganisation von adaptivem Verhalten.
VDI Fortschrittberichte, Reihe 8, Nr. 663, Ilmenau 1997, pp. 99-108.

Task-based representation in lifelong learning incremental neural networks*

Fred Henrik Hamker, Horst-Michael Gross

Technische Universität Ilmenau
Fachgebiet Neuroinformatik
D-98684 Ilmenau, Germany
http://cortex.informatik.tu-ilmenau.de/~fred
e-mail: fred@informatik.tu-ilmenau.de

ABSTRACT

A lifelong learning incremental neural network based on the Growing Neural Gas is presented. The Growing Neural Gas inserts new nodes depending on a local error measure and thus, the representation of input patterns depends on the fulfillment of the actual task, which is defined as a task based representation. In extension, our algorithm learns to insert by evaluating the course of the error in an insertion-evaluation cycle. On an artificial data set, it is demonstrated that the algorithm stops insertion inside of overlapping decision areas and adapts to changing environments while preserving old prototype patterns.

1. Introduction

Learning is one of the main issues of artificial neural network design. It describes a mechanism by which a system obtains a representation of its environment. Recent research addresses the topic of on-line learning, incremental learning and lifelong learning, which all discuss the same problem but emphasize different aspects. The necessity for on-line learning, in which the couplings of the network are updated after the presentation of each example, arises if not all training patterns are available all the time [8] [17]. Most publications referring to on-line learning focus on the role of the learning rule and the convergence of the learning process. For systems, like robots, which are faced with patterns during their entire lifetime, studying on-line learning in contexts such as a changing environment [17] encounters the problems of stability and plasticity. Incremental learning addresses the ability of repeatedly training a network with new data, without destroying the old prototype pattern. Lifelong learning emphasizes learning throughout the entire lifetime. Another approach in the context of lifelong learning, but not discussed here, is the transfer of knowledge from one task to another by learning task-independent knowledge [24].

* supported by the BMBF, Grant No. 413-5839-01 M 3014C - MIRIS-Project.

The questions of stopped learning: Is the collected data relevant and sufficient to cope with the problem? and When should the learning process be stopped?, changes in the context of lifelong learning to the questions: Is the network flexible enough to learn new data? and Does it remember previous data? The aim of this approach is to reveal new lifelong learning algorithms, based on incremental neural networks with a local insertion criterion and to develop sophisticated neural systems, which learn throughout their lifetime and automatically adapt to changing environments without immediately forgetting previous information.

2. Task-based representation

Unsupervised learning algorithms like Kohonen's Self-Organizing Feature Maps (SOFM) [18] or Neural Gas [19] freeze the ability of learning by decaying parameters. Networks with a global representation of knowledge, like a Multi-Layer-Perceptron (MLP) have to suffer from the disadvantage that changing only one weight affects nearly all patterns stored in the network. Furthermore, the weights are significantly under-constrained because of the exclusively error-driven learning method [21]. Networks with a local or distributed representation of knowledge appear to be better candidates for lifelong learning scenarios. One type of a local representation of knowledge utilized in recent literature for on-line learning are Radial Basis Function networks (RBF) [8]. Nevertheless, RBF have a fixed number of nodes, no matter whether the weights and the width of the Gaussians of the hidden nodes are determined by a previous clustering method or by on-line training [8]. Since the data stream is open and not known in advance, the restriction to a fixed number of nodes can lead to a suboptimal representation. Thus, lifelong learning demands changes of parameters independent of time and a pattern memory independent of the storage of a new pattern.

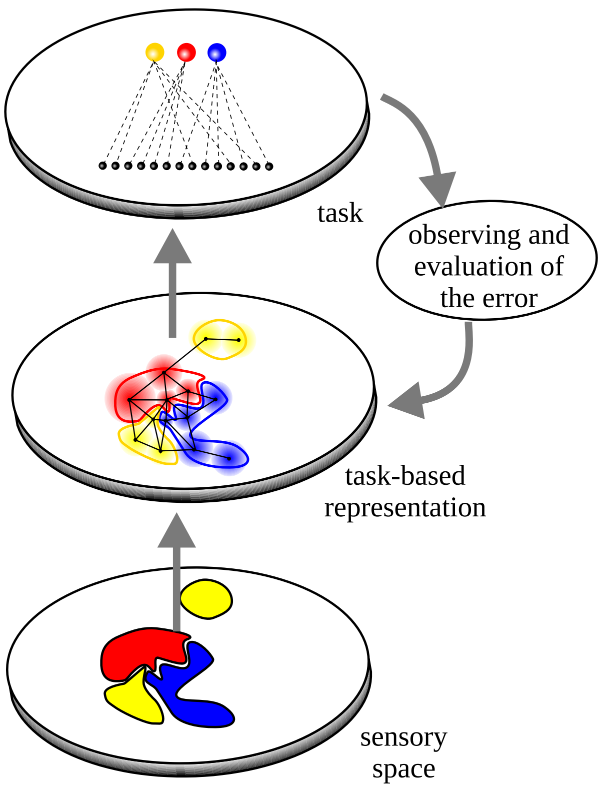


Fig. 1: Task-based representation of the sensory space.

Incremental networks with a local representation of patterns fulfill these demands. The Adaptive Resonance Theory (ART) [14] is aimed at learning new associations without forgetting old ones. ART networks, Fuzzy-ARTmap [5] and similar networks like CLAN [16] 'insert' new nodes based on a similarity measure. The other family of incremental networks use an error measure to insert new nodes. They can be subdivided into local error insertion rules like Growing Cell Structures [9], Growing Neural Gas [10], Dynamic Cell Structures [3] and global error insertion rules like Cascade-Correlation [7] or an incremental one hidden layer Perceptron [6]. To evaluate some incremental networks in the supervised case, we performed a benchmark on Fuzzy-ARTmap, Growing Cell Structures and Growing Neural Gas in comparison to a MLP [15]. Although the performance depends on the data set, the Growing Cell Structures and Growing Neural Gas perform not worse than a MLP, but they converge faster and parameter changes show only a little effect on their results. In most cases, the performance of the Fuzzy-ARTmap is not as good as that of the other networks [15]. The Cascade-Correlation algorithm performed worse and required one order of magnitude more epochs than the GCS on the two spiral benchmark [9]. The incremental learning technique that constructs a single sigmoidal hidden layer is shown to outperform Cascade-Correlation, but further comparisons are needed [6].

Nevertheless, the most important question in lifelong learning incremental networks addresses the rule of insertion. Inserting new nodes solely in dependence on the similarity of the input pattern leads to a purely sensor-based representation, which does not reflect the requirements of further processing stages. In contrast, an error-based insertion adapts the representation in dependence on the task and therefore leads to a task-based representation (Fig. 1). A task-based representation is superior to a sensor-based representation, because for recognition as well as for action the representation changes in order to reduce the error of the whole system. Learning a sensor-based representation arranges the input pattern according to its similarity which is in general a useful way to decrease the amount of possible solutions [21] but it is not necessarily an optimal solution to solve the task [20].

On-line learning in a lifelong context is a stochastic process and the distribution of the input signals is unknown. Without reducing the plasticity of the network by modifying the learning rates, fluctuations of the weights are inevitable. As explained, a time-dependent decrease of learning parameters is not applicable to lifelong learning. Amari proposed the learning of a learning rule [2]. According to his proposal, the change of the weight vector reveals whether it reached an optimal state. Heskes and Kappen [17] developed an algorithm for learning a parameter adaptation by estimating the statistics of the weights or by estimating the error potential of the algorithm. The latter requires the training of an ensemble of networks.

For the type of incremental networks with a local error insertion rule, we propose an error-based learning of the learning and insertion parameters by using the error information gained from the performance of the system on its task. This credit assignment approach has its origin in the original algorithm [9], but in contrast to the original insertion criterion the error has to be evaluated (Fig. 1).

3. Basic Principles

In general, a lifelong learning algorithm has to cope with complex boundaries or overlaps in the decision areas (Fig. 2) and the error of the network depends on the available data. Data with linearly separable decision areas should be learned without any error for all networks, which should also be achieved on data with complex boundaries. Theoretically, discrete overlaps can be solved, but with an unacceptable demand of resources. A continuous overlap can not be solved by any network. The boundary between the decision areas depends on the probability distribution of each class and should minimize the overall error.

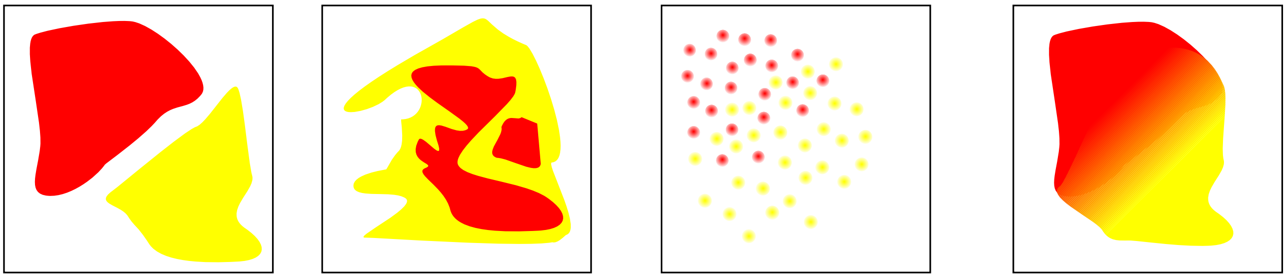


Fig. 2: Possible decision areas in feature space. A decision area is an area in the feature space that requires the recognition of a specific class or the choice of a specific action. From left to right, the figure shows linearly separable boundaries, complex boundaries, a discrete overlap and a continuous overlap. The kind of boundary or overlap between decision areas affects the learning and insertion strategy: Linearly separable boundaries are easy to learn and the error will decrease to zero. Decision areas with complex boundaries require high separation capabilities but can be solved without error. A discrete overlap can also be solved, but in general with an unaccaptable demand of resources. A continuous overlap always leads to an error in solving the task.

In contrast to a limited data set with fixed boundaries, in lifelong learning the boundaries may change and new clusters may emerge. Nevertheless, although the boundaries change over time, the problems concerning the overlap of decision boundaries will persist. Thus, growth has to be stopped if a further insertion is not useful. For this reason, each node not only owns an averaged longterm error counter, it is also equipped with an insertion threshold and an averaged longterm error counter at the moment of the last insertion (insertion error) (Fig. 3). An insertion is only carried out, if the error is higher than the insertion threshold. The node with the highest difference between error and insertion threshold is chosen and the insertion threshold is increased only if the actual error is higher or equal to the insertion error. This means the last insertion is termed unsuccessful, if the error does not decrease and the node is punished by increasing the insertion threshold.

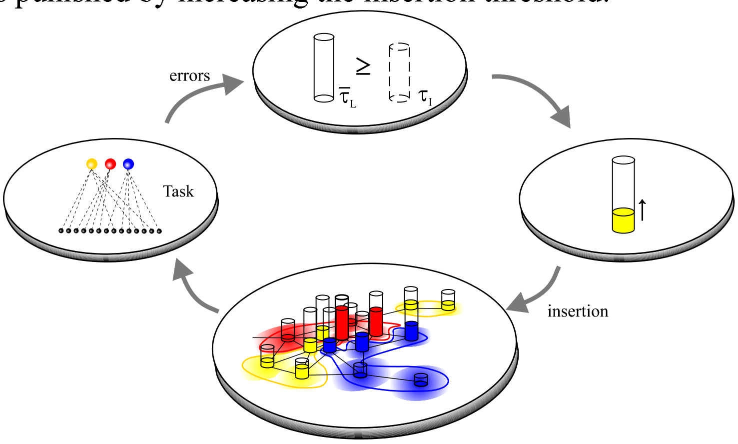


Fig. 3: Insertion evaluation cycle. The average long time error τ_L of the task is compared to the error at the moment of the last insertion τ_I. If this error is greater or equal, the insertion was not successful and the insertion threshold τ_ϑ is increased. If the threshold reaches the average error, a further insertion is not possible at that location. To permit exploration in futur, the threshold can be decreased with a large time

Another aspect in lifelong learning incremental networks concerns the adaptivity of the nodes. In [1] an error modulated Kohonen type learning rule was used to achieve a uniform approximation error independent of the input probability density. Here the modulation depends on the age of a node and on the difference between the average long time error and the average short time error and aims at reducing fluctuations. This means a node learns more if the short time error is larger than the long time error and if the node is younger.

In most applications at least, the amount of storage capacity and simulation time are limited. For this reason, a deletion criterion is introduced to remove redundant nodes. This can be useful, especially if the environment changes very often or the exploration rate is very high, e.g. the insertion threshold has a small time constant. The deletion criteria takes into account the number of neighbors, the similarity of the output weights and the size of the Voronoi region.

4. Lifelong learning scenarios

An autonomous agent is a typical example in which a system is faced with a changing environment, and thus, has no prior knowledge about the training set. Many authors in this field use vector quantization techniques like Kohonen's feature map, Neural Gas or Radial Basis Function networks for clustering the input data space [11] [12] [23], but passive clustering is not sufficient for an adequate sensory representation. To avoid this shortcoming, some authors proposed learning only in case of negative reinforcement signals [11], which according to our definition leads to a task-based representation. Nevertheless, incremental networks have the advantage that the number of nodes is also a result of learning by doing. The interaction of an animat with its environment causes reinforcement signals, which adapt the action selection as well as the sensory representation (Fig. 5). The output weights may be trained by a neural version of Q-learning [12] or other reinforcement learning algorithms. An incremental network used so far in the context of reinforcement learning is presented in [4], but in contrast to the lifelong adaptive network presented here, the nodes were restricted to a maximum amount.

Lifelong supervised learning seems confusing, because all the time an expert teaches a system (Fig. 4). But such a constellation can be biologically plausible by an inter-module supervision [22]. Furthermore it can be useful for robotic systems, especially when operating in non-standardized environments. They must be equipped with the highest possible flexibility to fulfill their task and with robustness towards changing process-variables for an economic employment, i.e., assembly tasks like error detection, flaw location and sorting. This kind of application demands a lot of flexibility when considering variable illumination and changing physical characteristics of the material, like deformation, dirtiness and wetness. The expert can be a human supervisor, who improves the system by teaching it again when necessary or in multisensor systems, a sub-system based on reliable but "expensive" sensors. The advantage of sorting systems based on this strategy is that the non-contacting and therefore fast visual sensor is responsible for separation and access initiation. The contacting tactile sensors are only needed for the verification and learning of the visual hypothesis after access of the manipulator [13].

The change of weights from the hidden to the output layer is calculated according to the delta rule as proposed in the publication of the original algorithm [9] [10].

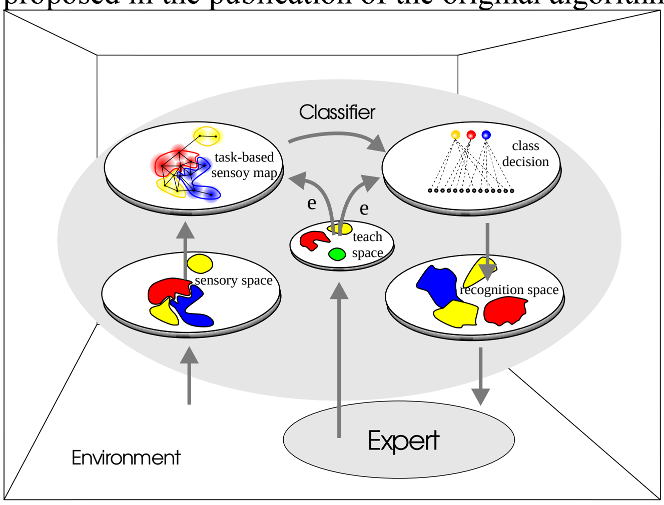


Fig. 4: An example of a lifelong supervised classifier.

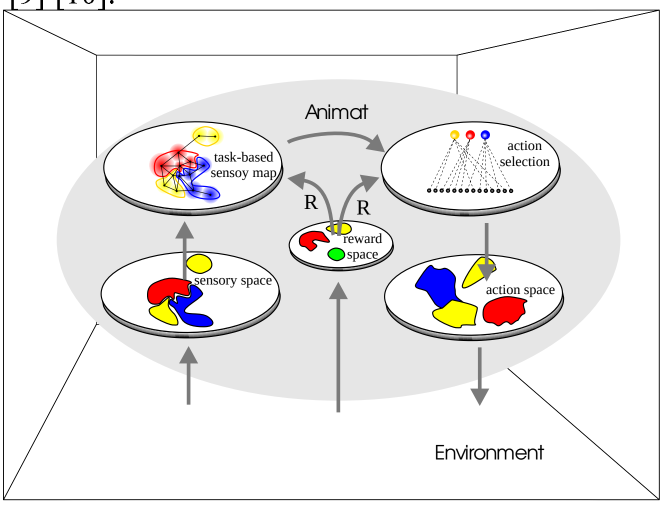


Fig. 5: An example of a simple agent with lifelong reinforcement learning.

6. Brief description of the algorithm

The basic principle is the same as in the original GNG [10]. Modifications concern the local counters of each node (Fig. 6), the control of learning and insertion and an explicit deletion criteria, which allows to steer the density of the nodes in consideration of their output-weight similarity. The following equations give a brief overview of the most important aspects.

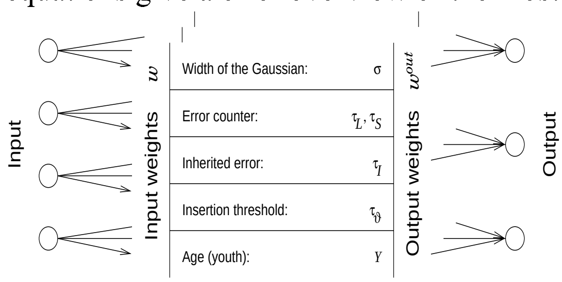


Fig. 6: Node of the lifelong GNG. Besides the width of the Gaussian each nodes owns a longtern error counter τ_L, a shortterm error counter τ_S, the inherited error at the moment of insertion τ_I, an insertion threshold τ_ϑ and the youth of the node Y, which decreases exponentially with the time constant T_Y from one to zero when the node was best matching. Despite the inherited error, which remains

- For all nodes i, calculate the Euclidian distance d_i of the input to the weight vector and locate the best matching unit b and calculate the activation of all nodes y_i with a Gaussian function (identical to [10]).
- Determine the output and the error $\Delta\tau$ according to the used supervised or reinforcement mechanism.
- Adapt the long time error counter τ_L and the short time error counter τ_S with the time constant T:

$$\tau_{(L/S)} := e^{-\frac{1}{T_{(L/S)}}} \cdot \tau_{(L/S)} + \left(1 - e^{-\frac{1}{T_{(L/S)}}}\right) \cdot \Delta\tau$$

- Decrease the insertion threshold τ_ϑ of each node and the youth Y of the best node b:

$$\tau_\vartheta := e^{-\frac{1}{T_\vartheta}} \cdot \tau_\vartheta \quad ; \quad Y_b := e^{-\frac{1}{T_Y}} \cdot Y_b$$

- Determine the quality measure of the best node and its neighbors for learning B^L and insertion B^I with consideration of the insertion threshold ϑ_{ins}.

$$B^L_{(b/n)} = \frac{\tau_{S_{(b/n)}} + 1}{\tau_{L_{(b/n)}} + 1} \quad ; \quad B^I_{(b/n)} = \tau_{L_{(b/n)}} - \tau_{\vartheta_{(b/n)}} \cdot (1 + \vartheta_{ins})$$

- Determine the learning rate of the best node and its neighbors from the quality measure and youth.

$$\eta'_{(b/n)_i} = \begin{cases} 0 & if \quad \alpha_i < 1 \\ \eta_{(b/n)} & if \quad \alpha_i > 2 \\ \alpha_i \cdot \eta_{(b/n)} & else \end{cases} \qquad \alpha_i = \frac{1 + B_i}{1 + \vartheta_L} + Y_i$$

  and allow a minimal learning rate of the input weights determined by ϑ_M:

$$\eta''_{(b/n)} = \max(\eta_M, \eta'_{(b/n)}) \quad ; \quad \eta_M = \eta'_{(b/n)} \cdot (1 - y) \cdot \vartheta_M$$

- Increase matching for b and its direct topological neighbors n.

$$\Delta w_b = \eta''_b \cdot (x - w_b)$$
$$\Delta w_n = \eta''_n \cdot (x - w_n) \quad \forall n \in N_b$$

- Adapt edges (identical to [10]).
- After $\lambda \cdot n_N$ steps, find node q and its neighbors f for insertion if the following criterion is fulfilled:

$$0 < K_{ins,q} = \max_{i \in G}(K_{ins,i}) \; ; \quad K_{ins,f} = \max_{i \in N_q}(K_{ins,i}) \; ; \qquad K_{ins,i} = B_i^I - Y_i$$

  - If q and f exist, insert new node r with the arithmetical average of weights and error counters.
  - Adapt moving insertion threshold for node r, q and f:

$$\text{IF} \quad \tau_L - \tau_I \cdot (1 - \vartheta_{ins}) > 0 \qquad \text{THEN} \quad \tau_\vartheta := \tau_\vartheta + \eta_\vartheta \cdot (\tau_L - \tau_\vartheta \cdot (1 - \vartheta_{ins}))$$

  - Check deletion criteria and find node d, whose criterion is higher than the deletion threshold ϑ_{del}:

$$\vartheta_{del} < K_{del,d} = \max_{i \in G}(K_{del,i}) \;\wedge\; \|N_d\| \geq 2 \;\wedge\; Y_d < \vartheta_{delY} \; ; \qquad K_{del,i} = \frac{1}{\|N_i\|} \cdot \frac{\overline{\Delta w}}{\bar{l}} \cdot \overline{\Delta w^{out}}$$

$$\overline{\Delta w} = \frac{1}{\|N_d\|} \sum_{j \in N_i} \|w_i - w_j\| \qquad \text{local density of the input weights}$$

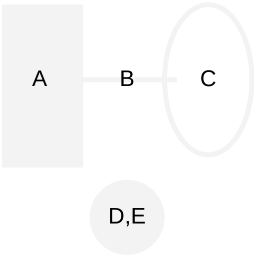
$$\bar{l} = \frac{1}{n_N} \sum_{j=1}^{n_N} \overline{\Delta w_j} \qquad \text{whole density of the input weights}$$

$$\overline{\Delta w^{out}} = \frac{1}{\|N_d\|} \sum_{j \in N_i} \|w_i^{out} - w_j^{out}\| \qquad \text{local density of the output weights}$$

In the above equations N_b denotes the set of direct topological neighbors of cell b and G the set of all cells of the network.

7. Examples

For a demonstration of the above ideas, we present a simulation of the lifelong supervised learning on an open data set containing overlappings i) without changing the environment and ii) with changing the environment. For illustration purposes a 2D artificial data set is chosen. It should be remarked that the performance of the presented algorithm in its converging stable state is comparable to the result of the original GNG on a public benchmark data set with real world data [15], as indicated in [25].
The first simulation illustrates the control of insertion, especially in overlapping areas. The overlap of the line with the ellipse in environment 1,4,5,6 are managed with only a few nodes. Also the algorithm exhibits no problems with the total overlap of the two circles in environment 3,4. Finally, the areas with low probability are represented as well.
The second simulation illustrates the performance while changing the environment to underline the flexibility of the algorithm. In some cases, the previously presented data is not completely kept (Fig. 11 env. 4). This depends on the deletion criteria. An area remains stable, if it is represented by three or more nodes, otherwise it may be destroyed by neighboring areas. Reducing the deletion threshold increases the density and ensures a saver representation but demands a higher simulation time and storage capacity for real data. In general, the algorithm shows a flexibility against unseen data, and keeps the old data in case of no contradiction.



	Class	Area	Environment (probability)					
			1	2	3	4	5	6
A	1	Rectangle	1	1	1	1	1	1
B	1	Line	1	0	0	1	1	1
C	2	Ellipse	1	1	1	1	1	1
D	3	circle	1	1	1	1	0.1	0.01
E	2	circle	0	0	1	1	0	0

Fig. 8: Artificial environments based on five areas (A-E). Each environment consists of different probabilities of each class in the areas.

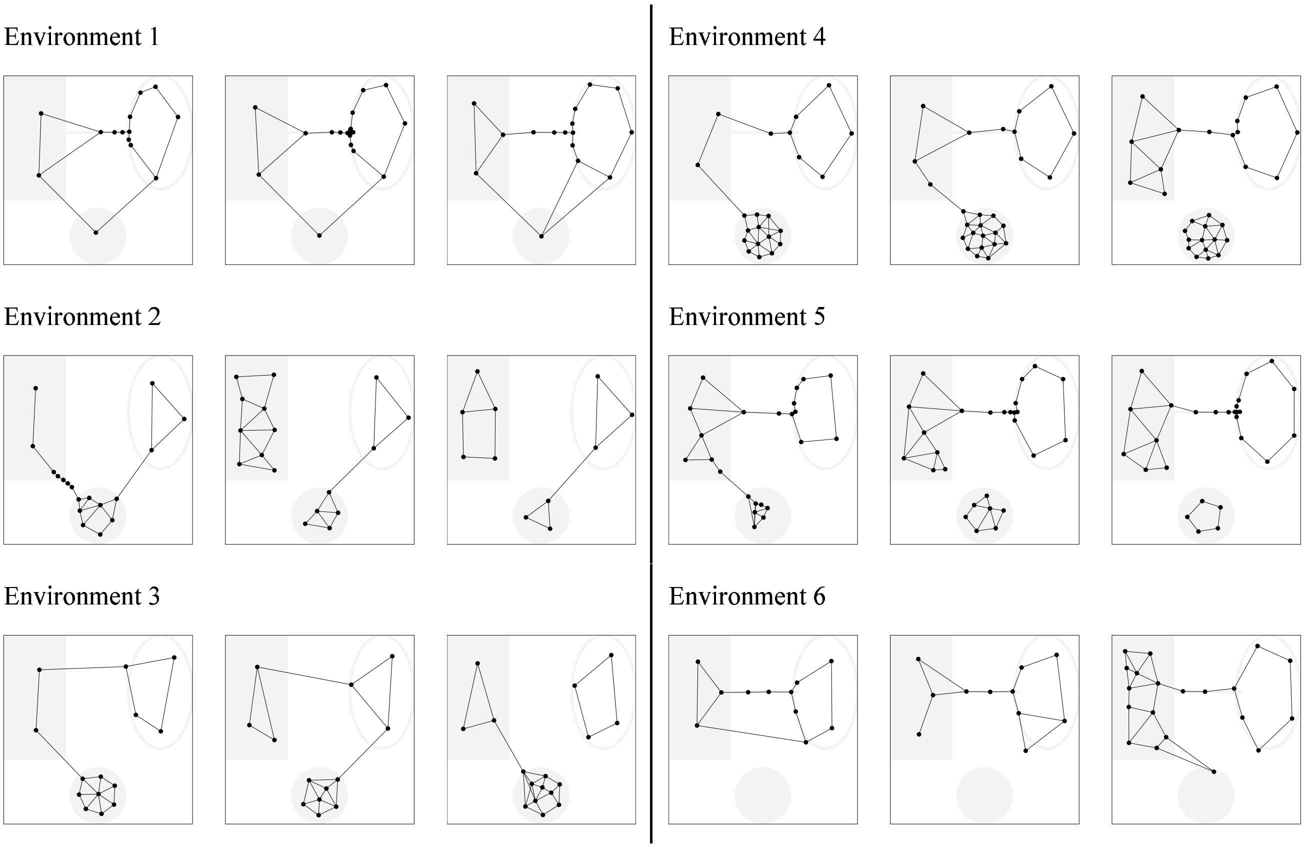
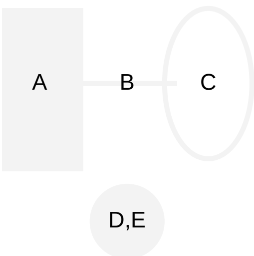


Fig. 9: Results of the algorithm on the artificial data set after 10000, 50000 and 200000 steps (adopted from [25]).



	Class	Area	Environment (probability)					
			1	2	3	4	5	6
A	1	Rectangle	1	1	0	1	1	0
B	1	Line	1	1	1	0	0	1
C	2	Ellipse	0	1	1	1	1	0
D	3	circle	0	0	0	0	1	1
E	2	circle	1	1	1	0	0	0

Fig. 10: Changing environment based on five areas (A-E). The environment changes from 1-6 always after 20000 steps.

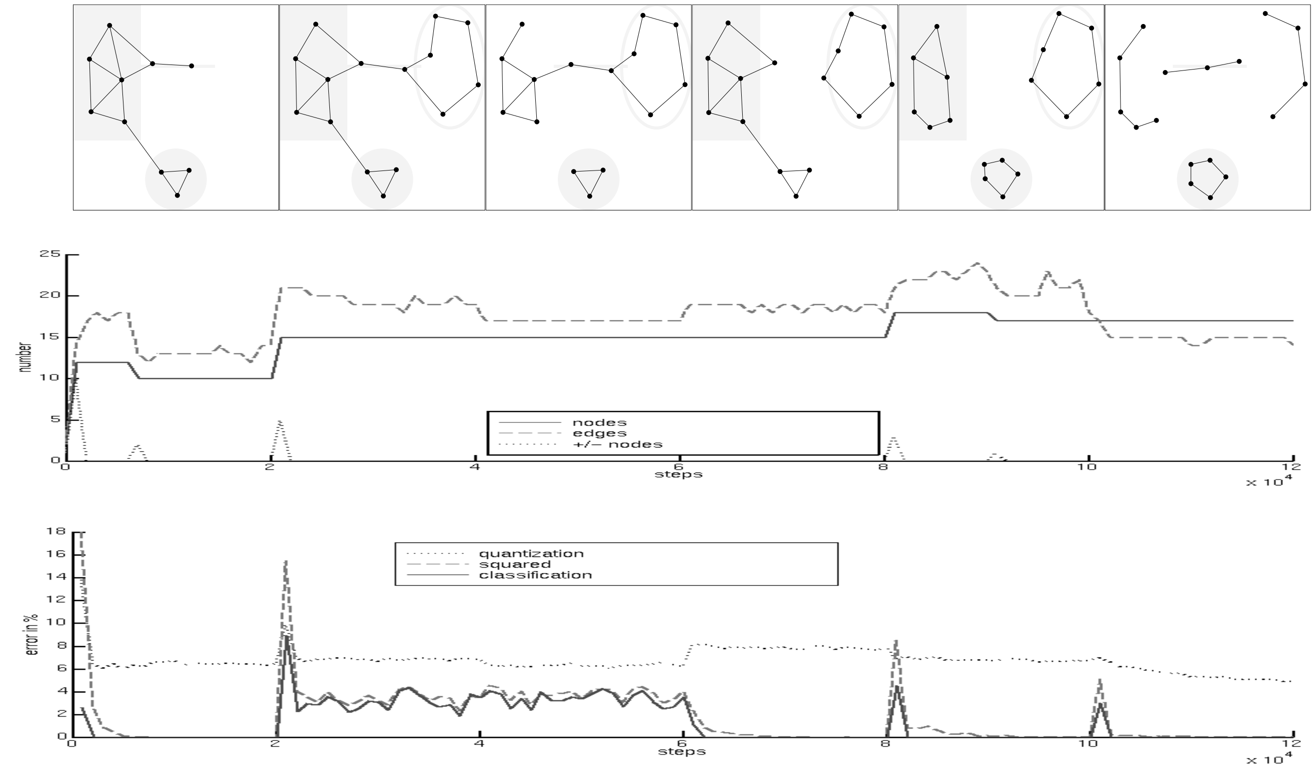


Fig. 11: Results of the algorithm on the changing environment simulation. Top: The result after every 20000 steps in the current environment. Middle: The course of nodes and edges in the simulation. Bottom: The quantization, squared and classification errors. Although a permanent error in environment 2 and 3 occurs, the number of nodes does not increase. After changing the environment, the network quickly adapts to the new situation and preserves the old pattern, which is clearly visible in environments 3 and 6.

8. Conclusion

A lifelong learning incremental neural network based on the GNG algorithm was presented to coordinate insertion and learning. It was shown that the network can learn to stop insertion in regions where the error can not be decreased. Furthermore, in changing environments the network remains stable for old prototype patterns and adaptive for new or modified prototype patterns.
Although the presented algorithm is derived heuristically and lacks a theoretical basis, the results obtained on artificial as well as on real world data sets indicate a good performance and are a promising step towards lifelong learning in neural networks. A detailed performance evaluation is in work.

Acknowledgment

We thank T. Vesper for his extremely fruitful discussions and for implementing the algorithm in his diploma thesis. Furthermore, we thank D. Surmeli for his thoroughly proofreading.

References

[1] Ahrns, I.; Bruske, J.; Sommer, G.: *On-line learning with Dynamic Cell Structures*. ICANN'95, pp. 141-146, 1995.

[2] Amari, S.: *A theory of adaptive pattern classifiers*. IEEE Transactions on Electronic Computers, vol. 16 (1967), pp. 299-307.

[3] Bruske, J.; Sommer, G.: *Dynamic cell structure learns perfectly topology preserving map*. Neural Computation, vol. 7 (1995), pp. 845-865.

[4] Bruske, J.; Ahrns, I.; Sommer, G.: *Neural Fuzzy control based on reinforcement learning and Dynamic Cell Structures*. EUFIT'96, Aachen 1996.

[5] Carpenter, G. A.; Grossberg, S.; Markuzon, N.; Reynolds, J. H.; Rosen, D. B.: *Fuzzy ARTMAP: A Neural Network Architecture for Incremental Supervised Learning of Analog Multidimensional Maps*. IEEE Trans. On Neural Networks, vol. 3 no 5 (1992), pp. 698-713.

[6] Dunkin, N.; Shawe-Taylor, J.; Koiran, P.: *A new incremental learning technique*. In:Neural Nets, WIRN Vietri-96, Proceedings of the 8th Italian Workshop an Neural Nets. Springer, pp. 112-118, 1997.

[7] Fahlman, S. E.; Lebiere, C.: *The cascade-correlation learning architecture*. In: Advances in Neural Information Processing Systems 2. Morgan Kaufmann Publishers, pp. 524-532.

[8] Freeman, J. A. S.; Saad, D.: *On-line learning in radial basis function networks*. Neural Computation (1996), submitted.

[9] Fritzke, B.: *Growing cell structures – A self-organizing network for unsupervised and supervised learning*. Neural Networks, vol. 7 no 9 (1994), 1441-1460.

[10] Fritzke, B.: *A growing neural gas network learns topologies*. Advances in Neural Information Processing Systems, vol. 7 (1995).

[11] Gaussier, P.; Zrehen, S.: *A topological neural map for on-line learning: Emergence of obstacle avoidance in a mobile robot*. From animals to animats 3: Proceedings of the Third International Conference on Simulation of Adaptive Behavior, MIT Press, pp. 282-290, 1994.

[12] Gross, H.-M; Stephan, V.; Boehme, H.-J.: *Sensory-based robot navigation using self-organizing networks and Q-learning*. WCNN'96, World Congress on Neural Networks, San Diego, pp. 94-99, 1996.

[13] Gross, H.-M; Hamker, F.; Debes, K.: *Visuelle Wahrnehmung im Handlungskkontext - Ein Sehsystem als aktiv handelnder Agent*. IWK 95: Proc of 40. Int. Col. Ilmenau , pp. 731-737, 1995.

[14] Grossberg, S.: *Adaptive pattern classification and universal recoding: I. Parallel development and coding of neural feature detectors*. In: Biological Cybernetics, vol. 23 (1976), pp. 121-134.

[15] Hamker, F.; Heinke, D.: *Implementation and Comparison of Growing Neural Gas, Growing Cell Structures and Fuzzy Artmap*. Schriftenreihe des FG Neuroinformatik der TU Ilmenau, Report 1/97, ISSN 0945-7518, 1997.

[16] Hartmann, G.: *The closed loop antagonistic network (CLAN)*. In: Elsevier Science Publishers B. V. Ed.: Eckmiller, R. North-Holland, pp. 279-285, 1990.

[17] Heskes, T. M.; Kappen, B.: *On-line learning processes in artificial neural networks*. Mathematical Foundations of Neural Networks. Elsevier, Amsterdam, pp. 199-233, 1993.

[18] Kohonen, T.: *Self-organized formation of topologically correct feature maps*. Biological Cybernetics, 43 (1982), pp. 59-69.

[19] Martinez, T. M..; Schulten, K. J.: *A "neural gas" network learns topologies*. ICANN'91, pp. 397-402..

[20] Möller, R.; Groß, H.-M.: *Sensorische Repräsentationen - Schwachstelle von Wahrnehmungsmodellen*. In: Proc. of Workshop "Artificial Life", GMD, Sankt Augustin, 1995.

[21] O'Reilly, R. C.: *The LEABRA model of neural interactions and learning in the neocortex*. Dissertation Department of Psychology, Carnegie Mellon University, 1996.

[22] Phillips, W. A.; Singer, W.: *In search of common foundations for cortical computation*. Behavioral and Brain Sciences, to appear 1997.

[23] Salomon, R.: *Radial Basis Function networks for autonomous agent control*. Proceedings of the International Conference on Neural Networks (ICNN'97), to appear, 1997.

[24] Thrun, S.: *Explanation-based neural network learning: A lifelong learning approach*. Kluwer Academic Publishers, 1996.

[25] Vesper, T.: *Lebenslanges Lernen auf der Basis des Growing Neural Gas Algorithmus*. Diplomarbeit, FG Neuroinformatik der TU Ilmenau, 1997.