Incremental Unsupervised Time Series Analysis using Merge Growing Neural Gas

Andreas Andreakis, Nicolai v. Hoyningen-Huene, and Michael Beetz

Technische Universität München, Intelligent Autonomous Systems Group,
Boltzmannstrasse 3, 85747 Garching, Germany
{andreaki,hoyninge,beetz}@cs.tum.edu

Abstract. We propose Merge Growing Neural Gas (MGNG) as a novel unsupervised growing neural network for time series analysis. MGNG combines the state-of-the-art recursive temporal context of Merge Neural Gas (MNG) with the incremental Growing Neural Gas (GNG) and enables thereby the analysis of unbounded and possibly infinite time series in an online manner. There is no need to define the number of neurons a priori and only constant parameters are used. MGNG utilizes a rather unknown entropy maximization strategy to control the creation of new neurons in order to focus on frequent sequence patterns. Experimental results demonstrate reduced time complexity compared to MNG while retaining similar accuracy in time series representation.

Key words: time series analysis, unsupervised, self-organizing, incremental, recursive temporal context

1 Introduction

Time series represent the vast majority of data in everyday life and their handling is a key feature of all living creatures. Automatic processing of sequential data has received a broad interest e.g. in action recognition, DNA analysis, natural language processing or CRM systems, to name just a few.

The analysis of time series aims at two main goals, namely the identification of the nature of the underlying phenomenon given observed example sequences and the forecasting of future values. Both of these goals depend upon a good representation of patterns in observed time series data. One approach, that has proven to be successful, is to build a quantization of the time series data to form a compact representation in an unsupervised way.

We propose Merge Growing Neural Gas (MGNG) as a recursive growing self-organizing neural network for time series analysis. The state-of-the-art Merge Neural Gas (MNG) [1] is extended to an incremental network by utilizing Growing Neural Gas (GNG) [2]. The theoretically founded recursive temporal context of MNG represents the input history as an exponentially decayed sum and is inherited for MGNG. Instead of MNG's entropy maximization a rather unknown strategy is used which avoids instabilities during training.

MGNG allows for online clustering of time series in previously unknown and possibly infinite data streams by the use of constant parameters only. There's no need to know the number of neurons in advance due to the growing of the network. The proposed algorithm exhibits faster runtime performance than MNG while keeping similar accuracy in the time series representation.

The remainder of the paper is organized as follows: After a review of related work in unsupervised time series analysis in Section 2, the MGNG algorithm is explained in Section 3 in detail. We evaluate the performance of the proposed approach in three experiments comparing our results with MNG and other models in Section 4. Conclusions and future prospects round off the work.

2 Related Work

Based on well known unsupervised models like the Self Organizing Map (SOM) [3] and Neural Gas (NG) [4] several extensions have been proposed for sequential input data. Common approaches use hierarchies [5], non-Euclidean sequence metrics [6], time-window techniques [4] or mapping to spatial correlations [7] and there exist a wider field of recursive models [8].

Recursive sequence models extend unsupervised neural networks by recursive dynamics such as leaky integrators [9]. Hammer et al. give an overview over recursive models [8] and present an unifying notation [10]. Temporal Kohonen Map (TKM) [11], Recurrent SOM (RSOM) [12], Recursive SOM (RecSOM) [13], SOM for structured data (SOM-SD) [14], Merge SOM (MSOM) and Merge Neural Gas (MNG) [1] represent popular recursive models which have been applied in several applications [15–19]. The specific models differ mainly in their internal representation of time series, which influences the capacity of the model, the flexibility with respect to network topology and processing speed. MNG has shown superior performance to the other recursive models for acceptable time complexity.

All extensions towards quantization of temporal data have in common that the optimal number of neurons has to be known in advance. However, too many neurons waste resources and may lead to overfitting and too less neurons cannot represent the input space well enough and might therefor lead to high quantization errors. Beside this holds also for non-temporal base models, time series exacerbate the uncertainty in the correct number before the analysis because of the combinatorial explosion of sequences.

Growing Neural Gas (GNG) was introduced by Fritzke [2] as an incremental unsupervised neural network for non-temporal data. Starting with two neurons GNG grows in regular time intervals up to a maximal size. Connections between neurons are created by topology preserving Competitive Hebbian Learning. Only the best matching neuron and its direct topological neighbors are updated for each input signal leading to lower time complexity than SOM or NG where the whole network is updated. All used learning parameters are constant and enable the handling of infinite input streams. Also the variant GNG-U has been introduced for non-stationary data distributions [20].

Kyan and Guan proposed Self-Organized Hierarchical Variance Map (SO-HVM) in [21] as another incremental network that unfortunately lacks of the capability for online processing through declining parameters and shows beside better accuracy with a higher complexity than GNG.

To the best of our knowledge no growing model which utilizes recursive dynamics and their associated advantages has been published, yet.

3 Merge Growing Neural Gas

Merge Growing Neural Gas transfers features of GNG into the domain of time series analysis by utilizing the state-of-the-art temporal dynamics of MNG.

MGNG is a self-organizing neural network consisting of a set of neurons $\mathcal{K}$ representing sequences connected by edges $\mathcal{E}$. Each neuron $n \in \mathcal{K}$ comprises of a weight vector $\mathbf{w}_n$ representing the current time step and a context vector $\mathbf{c}_n$ representing all past time steps of a sequence, both having the dimensionality of the input space.

An input sequence $\mathbf{x}_1, \ldots, \mathbf{x}_t$ is assigned to the best matching neuron (also called winner or BMU) by finding the neuron n with lowest distance d_n:

$$d_n(t) = (1-\alpha) \cdot \|x_t - \mathbf{w}_n\|^2 + \alpha \cdot \|\mathbf{C}_t - \mathbf{c}_n\|^2 \qquad (1)$$

The parameter $\alpha \in [0,1]$ weights the importance of the current input signal over the past. $\mathbf{C}_t$ is called the global temporal context and is computed as a linear combination (merge) of the weight and context vector from the winner r of the time step $t-1$:

$$\mathbf{C}_t := (1-\beta) \cdot \mathbf{w}_r + \beta \cdot \mathbf{c}_r \qquad (2)$$

The parameter $\beta \in [0,1]$ controls the influence of the far over the recent past. The temporal context $\mathbf{C}_t$ constitutes an exponentially decayed sum of all past winner's weight vectors beside the current. As proven by Strickert and Hammer [1] it converges to the optimal global temporal context vector $\mathbf{C}_t^{opt}$ that can be written as:

$$\mathbf{C}_t^{opt} = \sum_{j=1}^{t-1} (1-\beta) \cdot \beta^{t-1-j} \cdot x_j \qquad (3)$$

The training algorithm for MGNG is depicted in Figure 1. Hebbian learning takes place by adapting the winner neuron and its neighbors towards the recent input signal x_t and the past $\mathbf{C}_t$ for given learning rates ϵ_w and ϵ_n (see line 11). The connection between the best and second best matching unit is created or refreshed following a competitive Hebbian learning approach (see line 8 and 9). All other connections are weakened and too infrequent connections are deleted depending on γ (see line 12 and 13). The network grows at regular time intervals λ up to a maximal size θ by insertion of new nodes based on entropy maximization (see lines 15a-e).

3.1 Entropy Maximization

In time series analysis we are usually interested in a representation of frequent sequence patterns. This is achieved by an entropy maximization strategy for node insertion because the entropy of a network is highest if the activation of all neurons is balanced. At high entropy more neurons are used for frequent sequences reducing the representation capacity for rare ones. This helps to focus on quantization of important information beside the usually combinatorial explosion for time series.

Following a rather unknown strategy of Fritzke [22] we insert a new node in regions with high activation frequency leading to an increase of the entropy of the network. Frequency is tracked by a counter of every neuron that is incremented every time the neuron is selected as winner (see line 10). New nodes are inserted between the most active neuron q and its most frequent topological neighbor f reducing the likelihood of both nodes q and f to be selected as winner and therefor increasing the overall entropy of the network. The new node l is initialized as the mean of the two selected nodes and inserted in-between them. The counters of q and f are reduced to reflect the expected decrease of activation while the new neuron takes over this activation. The parameter δ controls the amount of the activation shift (see lines 15a-e). All counters are subject to exponential decay by η to give recent changes a higher relevance (see line 16). To further increase the entropy nodes with no connections are deleted because the last selection as first or second best matching unit was too long ago (see line 14).

In the entropy maximization strategy of MNG [1] the parameter α of the distance function (see eq. (1)) is gradually increased and decreased based on the entropy of the network describing a zigzag curve with diminishing amplitude. This results in a training based alternately on weight and context vectors and unfortunately causes a temporary destabilization of the network for extreme values of α close to one, because winner selection is based on the past only omitting the recent time step of a sequence. Our entropy maximization strategy avoids these problems and provides online capability by neglecting the configuration of the input count and requiring constant parameters only.

4 Experimental Results

Three experiments were conducted to evaluate MGNG's performance in temporal quantization, density estimation and representation capacity.

4.1 Mackey Glass

The Mackey Glass time series is a 1-dimensional, continuous and chaotic function defined by the differential equation $\frac{dx}{d\tau} = -0.1x(\tau) + \frac{0.2x(\tau-17)}{1+x(\tau-17)^{10}}$ and is commonly used to evaluate the temporal quantization of recursive models [13, 1, 23].

The experiment was conducted on the models SOM, NG, GNG, MNG and MGNG which were trained with a time series of $150,000$ elements. After training,

```
 1. time variable $t := 1$
 2. initialize neuron set $\mathcal{K}$ with 2 neurons with counter $e := 0$ and random
    weight and context vectors
 3. initialize connections set $\mathcal{E} \subseteq \mathcal{K} \times \mathcal{K} := \emptyset$
 4. initialize global temporal context $\mathbf{C}_1 := \mathbf{0}$
 5. read / draw input signal $\mathbf{x}_t$
 6. find winner $r := \arg\min_{n \in \mathcal{K}} d_n(t)$
    and second winner $s := \arg\min_{n \in \mathcal{K} \setminus \{r\}} d_n(t)$
    where
          $d_n(t) = (1 - \alpha) \cdot \|\mathbf{x}_t - \mathbf{w}_n\|^2 + \alpha \cdot \|\mathbf{C}_t - \mathbf{c}_n\|^2$
 7. $\mathbf{C}_{t+1} := (1 - \beta) \cdot \mathbf{w}_r + \beta \cdot \mathbf{c}_r$
 8. connect $r$ with $s$: $\mathcal{E} := \mathcal{E} \cup \{(r, s)\}$
 9. $age_{(r,s)} := 0$
10. increment counter of $r$: $e_r := e_r + 1$
11. update neuron $r$ and its direct topological neighbors $\mathcal{N}_r$:
          $\mathbf{w}_r := \mathbf{w}_r + \epsilon_b \cdot (\mathbf{x}_t - \mathbf{w}_r)$ and $\mathbf{c}_r := \mathbf{c}_r + \epsilon_b \cdot (\mathbf{C}_t - \mathbf{c}_r)$
          $\mathbf{w}_n := \mathbf{w}_n + \epsilon_n \cdot (x_t - \mathbf{w}_i)$ and $\mathbf{c}_n := \mathbf{c}_n + \epsilon_n \cdot (\mathbf{C}_t - \mathbf{c}_i)$ $(\forall n \in \mathcal{N}_r)$
12. increment the age of all edges connected with $r$
          $age_{(r,n)} := age_{(r,n)} + 1 \quad (\forall n \in \mathcal{N}_r)$
13. remove old connections $\mathcal{E} := \mathcal{E} \setminus \{(a, b) | age_{(a,b)} > \gamma\}$
14. delete all nodes with no connections.
15. create new node if $t$ mod $\lambda = 0$ and $|\mathcal{K}| < \theta$
      a. find neuron $q$ with the greatest counter: $q := \arg\max_{n \in \mathcal{K}} e_n$
      b. find neighbor $f$ of $q$ with $f := \arg\max_{n \in \mathcal{N}_q} e_n$
      c. initialize new node $l$
          $\mathcal{K} := \mathcal{K} \cup \{l\}$
          $\mathbf{w}_l := \frac{1}{2} (\mathbf{w}_q + \mathbf{w}_f)$
          $\mathbf{c}_l := \frac{1}{2} (\mathbf{c}_q + \mathbf{c}_f)$
          $e_l := \delta \cdot (e_f + e_q)$
      d. adapt connections: $\mathcal{E} := (\mathcal{E} \setminus \{(q, f)\}) \cup \{(q, n), (n, f)\}$
      e. decrease counter of $q$ and $f$ by the factor $\delta$
          $e_q := (1 - \delta) \cdot e_q$
          $e_f := (1 - \delta) \cdot e_f$
16. decrease counter of all neurons by the factor $\eta$:
          $e_n := \eta \cdot e_n \quad (\forall n \in \mathcal{K})$
17. $t := t + 1$
18. if more input signals available goto step 5 else terminate
```

Fig. 1: Pseudocode for training of Merge Growing Neural Gas (MGNG).

the temporal quantization error [13] for up to 30 past time steps into the past was calculated by re-inserting the time series and saving the winner unit for each time step without modification of the network.

MGNG was configured using the following parameters: $\alpha = 0.5$, $\beta = 0.75$, $\theta = 100$, $\lambda = 600$, $\gamma = 88$, $\epsilon_w = 0.05$, $\epsilon_n = 0.0006$, $\delta = 0.5$, $\eta = 0.9995$. We used the same α and β for MNG setting the other parameters as described in [1]. Analogous parameters have been used for the non-temporal models SOM, NG and GNG, where in SOM a 10 to 10 rectangular lattice was used.

Figure 2 visualizes the temporal quantization error of the models for up to 30 time steps into the past.

MNG and MGNG have learned the temporal context and have a similar quantization performance. However MGNG required just 8,018s for the computation in contrast to 20,199s needed by MNG. The runtime advantage of MGNG is based on the underlying GNG model, which in comparison to NG requires less adaptions for each input signal.

Results for other recursive models such as RSOM, RecSOM and SOM-SD can be obtained from [9]. As expected they are superior to non-temporal models SOM, NG and GNG, but their performance lies below MNG's and MGNG's.

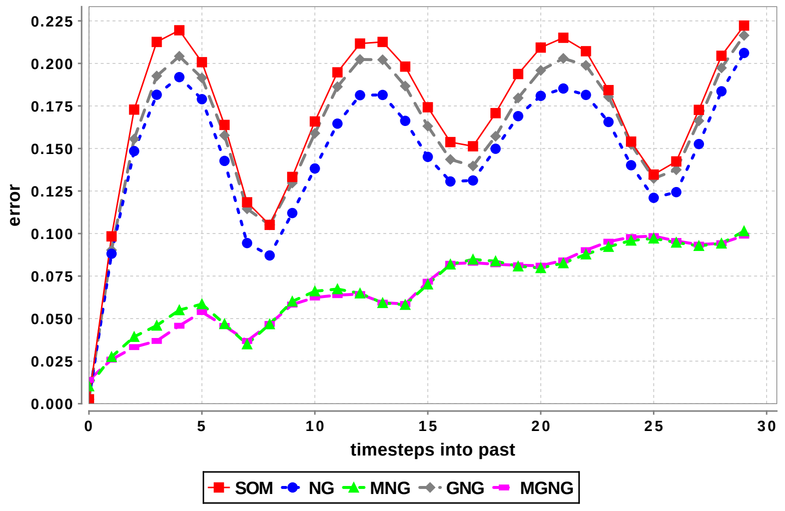

Fig. 2: Experimental results for the Mackey Glass time series.

4.2 Binary Automata

The Binary automaton experiment has been proposed by Voegtlin [13] in order to evaluate the representation capacity of temporal models using a Markov automaton with states 0 and 1 and the probabilities $P(0) = \frac{4}{7}$, $P(1) = \frac{3}{7}$,

$P(0|1) = 0.4$, $P(1|1) = 0.6$, $P(1|0) = 0.3$, $P(0|0) = 0.7$. A sequence with 10^6 elements was generated and trained in a network with 100 neurons. After training the winner units for the 100 most probable sequences are determined and for multiple winners only the longest sequence is associated. An optimal result would be achieved if each of the 100 sequences has an unique winner. The experiment was carried out with MNG and MGNG using $\alpha = 0.5$, $\beta = 0.45$ and the other parameters are inherited from the previous experiment.

MGNG shows a slight improvement against MNG in the representation capacity and a clear advantage in computation time. A total number of 64 longest sequences could be discriminated by MGNG requiring 69,143s and 62 sequences were discriminated by MNG in 131,177s.

However, both models cannot discriminate all 100 sequences, because recursive models that represent the temporal context as a weighted sum cannot discriminate between sequences with repeated '0' signals (such as: 0, 00, 0000...0). Choosing other values like '1' and '2' would improve the results.

4.3 Noisy Automata

The noisy automaton experiment originates from Hammer et al. [23] and its goal is to evaluate the density estimating capabilities of a temporal model by reconstructing the transition probabilities of a second order Markov model. Two-dimensional input signals are generated from three normal distributions with means $a = (0, 0)$, $b = (1, 0)$ and $c = (0, 1)$ and common standard deviations σ. Figure 3 visualizes the Markov automaton where the transition probabilities can be configured with the parameter x.

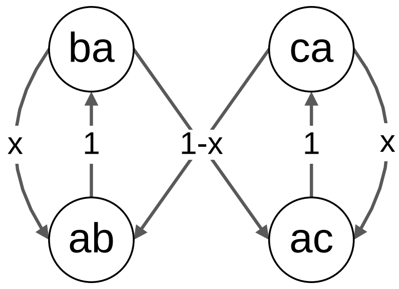

Fig. 3: Transition probabilities of the noisy automaton.

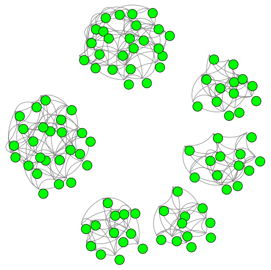

Fig. 4: Kamada-Kawai based MGNG Topology for $x = 0.5$

In the first part, the experiment was carried out with different transition probabilities $x \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8\}$ and with a constant $\sigma = 0.1$ using a total count of 10^6 input signals. In the second part $x = 0.4$ was set constant while different $\sigma \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$ were used. MNG was

(a) MNG varying x and σ separately.

x	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.4			
σ	0.1									0.2	0.3	0.4	0.5
$P(a\|ba)$	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.203	0.227
$P(b\|ba)$	0.0	0.11	0.211	0.31	0.399	0.513	0.595	0.687	0.791	0.471	0.312	0.515	0.248
$P(c\|ba)$	1.0	0.889	0.788	0.689	0.6	0.48	0.404	0.312	0.208	0.528	0.687	0.28	0.523
$P(a\|ca)$	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.122	0.0	0.0
$P(b\|ca)$	1.0	0.884	0.787	0.689	0.599	0.482	0.404	0.341	0.205	0.55	0.387	0.554	0.735
$P(c\|ca)$	0.0	0.115	0.212	0.31	0.4	0.517	0.595	0.658	0.794	0.449	0.49	0.445	0.264

(b) MGNG varying x and σ separately.

x	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.4			
σ	0.1									0.2	0.3	0.4	0.5
$P(a\|ba)$	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.216	0.409
$P(b\|ba)$	0.0	0.098	0.201	0.302	0.398	0.498	0.603	0.699	0.796	0.501	0.591	0.576	0.389
$P(c\|ba)$	1.0	0.901	0.798	0.697	0.601	0.501	0.396	0.3	0.203	0.498	0.408	0.207	0.201
$P(a\|ca)$	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.174	0.21	0.438
$P(b\|ca)$	1.0	0.9	0.798	0.7	0.6	0.498	0.398	0.3	0.2	0.587	0.557	0.34	0.145
$P(c\|ca)$	0.0	0.099	0.201	0.299	0.399	0.501	0.601	0.699	0.799	0.412	0.267	0.448	0.416

Table 1: Experimental results for the noisy automaton experiment.

configured with $\alpha = 0$ and $\beta = 0.5$ MGNG with $\alpha = 0.5$ and the same β. The other parameters are equal to the previous experiments.

The transition probabilities are reconstructed based on the backtracking method as described in [23]. As can be seen in Tables 1(a) and 1(b) the transition probabilities for both MNG and MGNG could be reconstructed accurately in the first part of the experiment. In the second part, MNG shows better results for $\sigma > 0.3$. However unlike MNG, MGNG is able to identify clusters without any further post-processing. Every subgraph in MGNG can be interpreted as a cluster and Figure 4 visualizes the six identified clusters using the Kamada-Kawai graph drawing algorithm [24], representing all possible sequence triples.

5 Conclusions

We introduced MGNG as a novel unsupervised neural network for time series analysis. Combining the advantages of growing networks and recursive temporal dynamics our model exhibits state-of-the-art accuracy in quantization, density estimation and representation of sequences in several experiments without requiring any a-priori knowledge in contrast to all other recursive networks. Our approach is simple to implement and shows better runtime performance than MNG. In future research we plan to investigate the extension of MGNG with different node creation and deletion strategies based on entropy and the maximal local variance to further improve the accuracy.

Acknowledgments. The work described in this paper was partially funded by the German Research Foundation DFG.

1. Strickert, M., Hammer, B.: Merge SOM for temporal data. Neurocomputing **64** (2005) 39–71
2. Fritzke, B.: A Growing Neural Gas network learns topologies. In Tesauro, G., Touretzky, D.S., Leen, T.K., eds.: NIPS. MIT Press (1995) 625–632
3. Kohonen, T.: Self-Organizing Maps. 3 edn. Springer, Berlin (2001)
4. Martinetz, T., Martinetz, T., Berkovich, S., Schulten, K.: 'Neural-gas' network for vector quantization and its application to time-series prediction. Neural Networks **4**(4) (1993) 558–569
5. Carpinteiro, O.A.S.: A Hierarchical Self-Organizing Map Model for Sequence Recognition. In: Proc. of ICANN. Volume 2. Springer, London (1998) 815–820
6. Hammer, B., Villmann, T.: Classification using non standard metrics. In: Proc. of ESANN. (2005) 303–316
7. Euliano, N.R., Principe, J.C.: A Spatio-Temporal Memory Based on SOMs with Activity Diffusion. In Oja, ed.: Kohonen Maps. Elsevier (1999) 253–266
8. Hammer, B., Micheli, A., Neubauer, N., Sperduti, A., Strickert, M.: Self Organizing Maps for Time Series. In: Proc. of WSOM, Paris, France (2005) 115–122
9. Hammer, B., Micheli, A., Sperduti, A., Strickert, M.: Recursive self-organizing network models. Neural Networks **17**(8-9) (2004) 1061–1085
10. Hammer, B., Micheli, A., Sperduti, A.: A general framework for unsupervised processing of structured data. In: Proc. of ESANN. Volume 10. (2002) 389–394
11. Chappell, G.J., Taylor, J.G.: The Temporal Kohonen map. Neural Networks **6**(3) (1993) 441–445
12. Koskela, T., Varsta, M., Heikkonen, J., Kaski, K.: Temporal sequence processing using Recurrent SOM. In: Proc. of KES, IEEE (1998) 290–297
13. Voegtlin, T.: Recursive Self-Organizing Maps. Neural Networks **15**(8-9) (2002)
14. Hagenbuchner, M., Sperduti, A., Tsoi, A.C.: A Self-Organizing Map for adaptive processing of structured data. Neural Networks **14**(3) (May 2003) 491– 505
15. Lambrinos, D., Scheier, C., Pfeifer, R.: Unsupervised Classification of Sensory-Motor states in a Real World Artifact using a Temporal Kohonen Map. In: Proc. of ICANN. Volume 2., EC2 (1995) 467–472
16. Farkas, I., Crocker, M.: Recurrent networks and natural language: exploiting self-organization. In: Proc. of CogSci. (2006)
17. Farka, I., Crocker, M.W.: Systematicity in sentence processing with a recursive Self-Organizing Neural Network. In: Proc. of ESANN. (2007)
18. Trentini, F., Hagenbuchner, M., Sperduti, A., Scarselli, F., Tsoi, A.: A Self-Organising Map approach for clustering of XML documents. In: WCCI. (2006)
19. Estevez, P.A., Zilleruelo-Ramos, R., Hernandez, R., Causa, L., Held, C.M.: Sleep Spindle Detection by Using Merge Neural Gas. In: WSOM. (2007)
20. Fritzke, B.: A self-organizing network that can follow non-stationary distributions. In: Proc. of ICANN. Springer (1997) 613–618
21. Kyan, M., Guan, L.: Local variance driven self-organization for unsupervised clustering. In: Proc. of ICPR. Volume 3. (2006) 421 – 424
22. Fritzke, B.: Vektorbasierte Neuronale Netze. PhD thesis, Uni Erlangen (1998)

23. Strickert, M., Hammer, B., Blohm, S.: Unsupervised recursive sequence processing. Neurocomputing **63** (January 2005) 69–97
24. Kamada, T., Kawai, S.: An algorithm for drawing general undirected graphs. Inf. Process. Lett. **31**(1) (1989) 7–15