Speech Dimensionality Analysis on Hypercubical Self-Organizing Maps

Panu Somervuo (panu.somervuo@hut.fi)
Helsinki University of Technology, Neural Networks Research Centre, P.O. Box 5400, FIN-02015 HUT, Finland

Abstract. The problem of finding the intrinsic dimension of speech is addressed in this paper. A structured vector quantization lattice, Self-Organizing Map (SOM), is used as a projection space for the data. The goal is to find a hypercubical SOM lattice where the sequences of projected speech feature vectors form continuous trajectories. The effect of varying the dimension of the lattice is investigated using feature vector sequences computed from the TIMIT database.

Keywords: dimension reduction, fractal dimension, multi-dimensional scaling, self-organizing map, sequence projection, speech analysis

1. Introduction

The common methodology for processing speech signals in pattern recognition applications is to compute feature vectors from fixed-length speech sample windows (speech frames) at constant time intervals. Usually the feature vectors are some kind of spectral representations of the signal. Speech dynamics can be investigated by observing their sequential order. Local temporal information can be stored into individual feature vectors by concatenating consecutive feature vectors or computing their time derivatives.

Since the feature vectors are points in a feature space and consecutive feature vectors form trajectories in that space, the entire speech data can be described as a graph, where the separate feature vectors are the nodes of the graph and the edges of the graph represent the trajectories of the speech. But each node of the graph need not represent only one feature vector. Feature vectors can be clustered and the nodes of the graph will then represent the prototypes of the resulting clusters.

In this work the adaptive prototypes of the feature vectors are located at the nodes of a structured lattice, the Self-Organizing Map (SOM) [12, 13]. The goal is to represent temporal feature sequences, speech trajectories, by means of a low-dimensional hypercubical SOM. The effect of varying the dimension of the SOM lattice in representing the feature vector sequences is investigated.

2. Experiments

TIMIT database [22] was used in the experiments. It consists of phonetically rich English sentences spoken by several speakers. Feature vectors were conventional 12-dimensional mel-scaled cepstral coefficient (MFCC) vectors weighted by a liftered sine. The mean vectors were subtracted from each sentence.

The goal of the work was to find a meaningful low-dimensional representation for *sequences* of speech feature vectors. Nevertheless, some methods for determining the intrinsic dimensionality of the data were applied also to separate feature vectors. These were the determination of the fractal dimension of the data set in Sec. 2.1 and data dimensionality reduction using multi-dimensional scaling in Sec. 2.2. When evaluating the quality of the SOMs in Sec. 2.3, the quantization error of data was first computed for separate feature vectors (as usually done) but then also when taking the temporal order of the feature vectors into account.

2.1. FRACTAL DIMENSION OF DATA SET

In order to roughly investigate the dimensionality of the data manifold, a subset of speech data was investigated by means of the fractal dimension measure proposed by Grassberger and Procaccia [9]. This measure, Correlation Dimension, is based on the pairwise distances of data points. Let $K(r)$ denote the correlation integral which measures the number of data pairs whose distance is smaller than a threshold r:

$$K(r) = \frac{1}{N(N-1)} \#\{(\mathbf{x}_i, \mathbf{x}_j) : \ ||\mathbf{x}_i - \mathbf{x}_j|| < r\}, \tag{1}$$

where # denotes the cardinality of a set and $N(N-1)$ is the number of all data pairs $(\mathbf{x}_i, \mathbf{x}_j)$. If $\log K(r)$ is plotted as a function of $\log r$, the fractal dimension of the data set can be read from the slope of the linear part of the curve.

Scaling of feature vector components affects the pairwise distances. This effect was removed from the data set by subtracting the mean of the data vectors and then scaling the variance of the vector components to unity. The Correlation Dimension was 5.4 for the data set which consisted of 25 randomly chosen TIMIT utterances (7770 12-dimensional feature vectors). The results were similar even when using different preprocessing methods. For 26-dimensional logarithmic mel-spectrum vectors, 12-dimensional cepstrum vectors with and without mean subtraction and with and without sine-liftering, the Correlation Dimension varied between 5.0 and 5.7. For comparison, 7770 12-dimensional ran-

dom vectors were picked from the uniform distribution and the normal distribution. For these data sets the Correlation Dimensions were 10.5 and 10.7. The higher the true fractal dimension of the data set is, the greater the size of the data sample should be in order to get an accurate value. Nevertheless, it can be quite safely assumed that the effective dimension of the current data set is not higher than six. It is worth noting that when determining the Correlation Dimension, the distances are computed in the entire feature space, not along the data manifold. Therefore, if the data set is folded, the true dimension of the data manifold may be actually smaller than suggested by this measure.

2.2. MULTI-DIMENSIONAL SCALING

Multi-dimensional scaling (MDS) methods, e.g. Sammon's mapping [20] or Curvilinear Component Analysis (CCA) [7] can also be used for finding the dimension of the data manifold. Different mappings can be compared by computing the mismatch s between the distances of data pairs in the original data space and in the projection space:

$$s = \sqrt{\frac{\sum_{i,j}(d_{ij}^o - d_{ij}^p)^2}{\sum_{i,j}(d_{ij}^o)^2}}, \tag{2}$$

where d_{ij}^o and d_{ij}^p denote the distances between items i and j in the original input space and in the projection space.

CCA was applied to the set of 1000 speech feature vectors. The dimension of the CCA projection space varied between 1 and 12 (the dimension of the feature vector). The mappings were initialized using the first M components of the data vectors when M was the dimension of the desired projection space. Alternatively, PCA could have been used by projecting the data vectors to the linear subspace spanned by the M largest principal components of the data set. After initialization, the mappings were adapted with 2000 iterations of the CCA algorithm [7]:

$$\Delta \mathbf{y}_j = \alpha(t) F(d_{ij}^p, \lambda) \frac{(d_{ij}^o - d_{ij}^p)}{d_{ij}^p} (\mathbf{y}_j - \mathbf{y_i}), \tag{3}$$

where $\mathbf{y}_j$ is the position vector of data item j in the projection space, $\alpha(t)$ is a gradient step, and $F(,)$ is a weighting function. Gradient step $\alpha(t)$ decreased linearly from 0.5 to zero during the iterations and F was a Gaussian (with value close to unity in the entire projection space in order to minimize the cost function in Equation 2).

The differences between the distances in the original feature space and in the CCA-projection space are shown in Figure 1. There is no

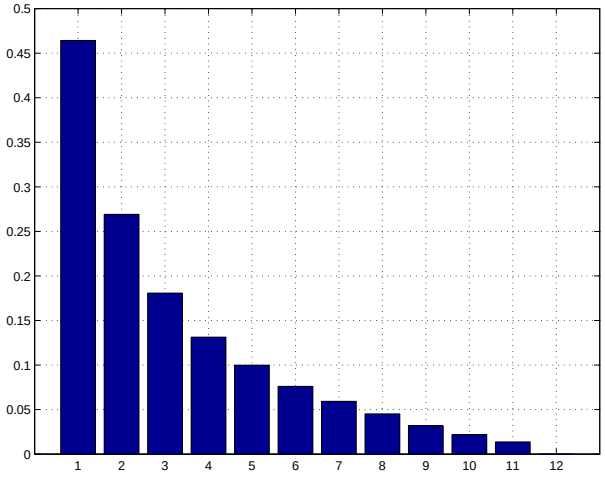

Figure 1. Data projections of 12-dimensional MFCC-feature vectors using Curvilinear Component Analysis. Error bars represent differences between data pair distances in the original feature space and in the projection space using Equation 2. Vertical axis represents the dimension of the projection space.

sharp change or plateau in the representation error which would indicate a good value for the low-dimensional projection of the data set. Instead, the representation error decreases smoothly as the dimension of the projection space increases. However, since the cost function includes the data pair distances in the original feature space, the noise of the feature vectors is also involved. The dimension of the effective data manifold may thus be smaller than the dimension of the projection space which most accurately preserves all distances between the original noisy data samples.

The current experiment was done using euclidean distances in the data space. Different results could be obtained by using curvilinear (or geodesic) distances, see e.g. [15].

2.3. SELF-ORGANIZING MAP

The Self-Organizing Map (SOM) [12, 13] is an adaptive, elastic grid which is fitted to the data by means of an unsupervised learning process. Each node of the grid contains a model representing the data. The structure of the grid can be freely chosen, but usually it is a regular, low-dimensional lattice. The main interest in this work was to investigate the effect of varying the dimension of the hypercubical lattice in representing the feature vector sequences.

Usually the models associated with the nodes of the SOM are discrete feature vectors which become representative prototypes of the

Table I. Sizes of hypercubical SOM lattices used in the experiments.

lattice dimension	number of the nodes in the lattice
2	32×31
3	$10 \times 10 \times 10$
4	$8 \times 5 \times 5 \times 5$
5	$4 \times 4 \times 4 \times 4 \times 4$
6	$4 \times 3 \times 3 \times 3 \times 3 \times 3$

data set during learning. In its basic formulation, the SOM algorithm organizes a VQ-codebook according to the similarity of separate feature vectors taking no temporal dependencies between the feature vectors into account. The batch training algorithm of the SOM can be expressed as [13]

$$\mathbf{m}_i = \frac{\sum_j h_{c(\mathbf{x}_j),i}(t)\mathbf{x}_j}{\sum_j h_{c(\mathbf{x}_j),i}(t)}, \tag{4}$$

where $\mathbf{m}_i$ is the model vector associated with the map unit i, $h_{c(\mathbf{x}_j),i}(t)$ is the neighborhood function, $c(\mathbf{x}_j)$ is the index of the best-matching unit (BMU) for data vector $\mathbf{x}_j$, and t is the time index. The neighborhood function controls the learning rates of the models on the map. Its shape determines the plasticity of the map and the degree of smoothness how the model vectors approximate the distribution of the data. In case of a Kronecker delta as a neighborhood function Equation 4 reduces to the LBG-algorithm [16].

Five different map lattices with approximately 1000 nodes in each were trained using the batch SOM algorithm, see Table 2.3. The hypercubical maps were initialized according to principal components of the training data. The Gaussian neighborhood function was used whose width (standard deviation) was half of the length of the largest lattice side in the beginning of training and it decreased linearly to 0.5 in the end of training. The training data consisted of 100 TIMIT sentences and the test data consisted of a different set of 100 TIMIT sentences.

2.3.1. *Normalization of the stiffness of the map lattice*
Since the number of the map nodes inside a fixed neighborhood radius increases as the map lattice dimension increases, a high-dimensional map lattice is more stiff (resulting in a higher quantization error) than a low-dimensional map. In order to compensate this effect, the value of the neighborhood function can be scaled according to the number of nodes belonging to the effective updating neighborhood. In case of a

hypercubical map lattice, in the end of the training when the effective width of the neighborhood contains only the nearest map neighbors, the data vectors belonging to the Voronoi regions of the nearest map neighbors of the given node can first be multiplied by the value of the original neighborhood function and then divided by the number of the nodes in the effective neighborhood (which is two times the lattice dimension in case of a hypercubical map). The effect of this is shown in Figure 2 (the plot with filled dots), the quantization error of the test data using maps with different lattice dimensions but the same number of nodes are very similar. Different map lattices with the same number of nodes yield mutually similar quantization errors also when the training is continued using the Kronecker delta as a neighborhood function, however, this may severely deteriorate the ordering of the map.

2.3.2. *Comparing different SOM lattices*
Quantization error measures the accuracy of the SOM in representing separate data vectors. In Figure 2, the quantization error was computed between data vectors and their BMUs without taking the temporal order of the data into account:

$$Q = \frac{1}{N} \sum_{j=1}^{N} ||\mathbf{x}_j - \mathbf{m}_{c(\mathbf{x}_j)}||^2. \quad (5)$$

Another way to compare different lattices is to measure the distance between the consecutive BMUs on the map. For topology-preserving maps two consecutive BMUs should be close each other (if the input sequence does not contain abrupt changes). This kind of measure has been earlier applied to static data [11]. Now instead of using those two nodes which are the best and second-best units for a single data vector, the two nodes are the BMUs for two consecutive data vectors in a time series. First the BMUs are found for data vectors and the standard quantization error (Equation 5) is computed. The shortest paths between each consecutive BMUs are then found and their lengths are added to this measure resulting in the following measure:

$$Q' = \frac{1}{N} \sum_{t=1}^{N} \left(||\mathbf{x}_t - \mathbf{m}_{c(\mathbf{x}_t)}||^2 + \min \sum_{i \in P} ||\mathbf{m}_{P(i)} - \mathbf{m}_{P(i+1)}||^2 \right), \quad (6)$$

where $P(i)$ is the ith map node index in the continuous path P from node $c(\mathbf{x}_t)$ to node $c(\mathbf{x}_{t+1})$. The shortest path between each node pair needs to be computed only once, and the result can be stored

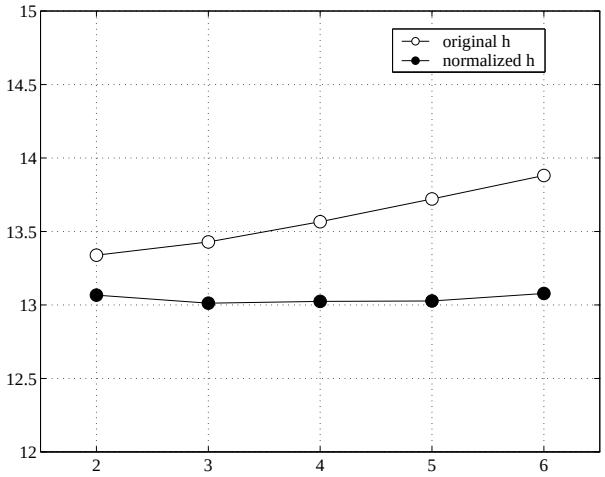

Figure 2. Quantization error of test data using hypercubical SOMs with lattice dimensions from 2 to 6. Different map lattices with the same number of nodes yield the same quantization error after the lattice stiffness normalization, see Sec. 2.3.1, h denotes the neighborhood function.

in a lookup table. Here Floyd's algorithm [1] was used for computing the shortest paths between all map nodes. The error measure Q' applied to different map lattices is shown in Figure 3. The quantization error decreases as the dimension of the map lattice increases since the higher-dimensional lattice has more node connections than a lower-dimensional one. The most distinguishable difference is observed between the two-dimensional map lattice and other maps. It can be also observed that the error measure increases as the map training is continued with a small value of neighborhood function. This is because the prototype vectors of the map are then spread more in the feature space and although the first term in the right hand side of Equation 6 decreases, the second term increases more.

In Figure 4, the quantization error is again computed between the data vectors and their BMUs, but now the BMUs are restricted to form continuous paths in the map. This means that the BMUs of two consecutive data vectors of the input sequence must be nearest neighbors in the map lattice. The best BMU sequence is found using dynamic programming.

Since the path of the BMUs must be continuous along the map lattice, the sampling rate of the data sequence must be sufficiently high so that if two ensuing phones in speech are represented at the opposite sides of a map lattice, the BMU trajectory is still able to reach both areas of the map. In this work the time resolution of the feature vectors was 1 ms.

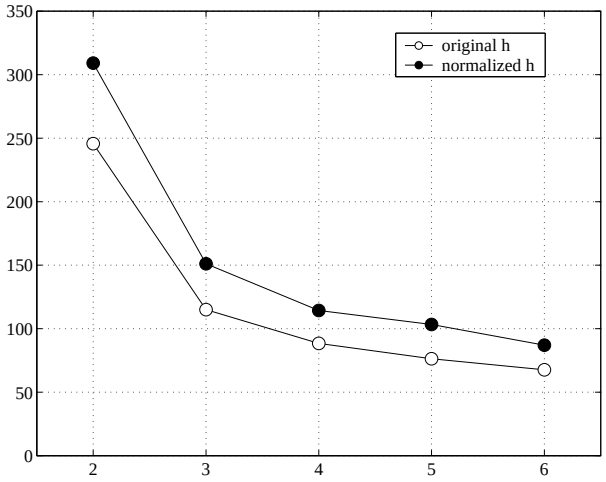

Figure 3. Quantization error of test data using hypercubical SOMs with lattice dimensions from 2 to 6 when the distances between the consecutive BMUs measured along the map lattice have been taken into account according to Equation 6.

When the quantization error was the same for all maps (after lattice stiffness normalization), the map lattices with the dimensions five and six gave the lowest quantization errors (the plot with filled dots in Figure 4). Since the five-dimensional lattice performs as well as the six-dimensional lattice but has less node connections, it can be concluded that the additional node connections in the higher-dimensional lattice are not used and thus the speech trajectories can be adequately represented by means of the five-dimensional hypercubical SOM.

2.3.3. *Experiments with growing SOMs and topographic product*
The Growing SOM (GSOM) algorithm [3] was also experimented for determining the proper dimensionality of the hypercubical SOM lattice. GSOM training begins using the map with only two nodes. The training proceeds then by repeating the conventional SOM training and the addition of new nodes in the lattice so that the dimension of the lattice may increase but its structure will always remain hypercubical. The results with GSOM were not very satisfactory, since the final dimension of the map lattice seemed to depend on the effective width of the neighborhood function. The expansion of the lattice dimension was more conservative if the neighborhood function was wide, but if it decreased during the training so that in the end it effectively contained only the nearest map neighbors, for some experimented data sets the final dimension of the map lattice could even exceed the dimension of the feature space. This happened despite a wide neighborhood function was used after each map lattice expansion.

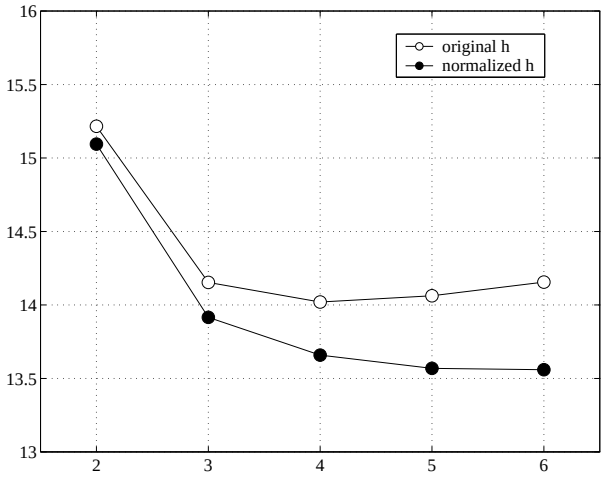

Figure 4. Quantization error of test data using hypercubical SOMs with lattice dimensions from 2 to 6 when the consecutive BMUs have been constrained to be nearest neighbors in the map lattice.

Topographic product [2] has been suggested for determining the proper dimensionality of the SOM. In this measure, data samples are not utilized at all. This means that temporal order of the feature vectors is not taken into account either. Main purpose of using this measure is to examine the folding of the map in the feature space. Topographic product Φ is defined as [2]:

$$\Phi = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j=i}^{N-1} \log \left(\prod_{k=1}^{j} \frac{d^o(\mathbf{m}_i, \mathbf{m}_{n_k^o(i)})}{d^o(\mathbf{m}_i, \mathbf{m}_{n_k^p(i)})} \frac{d^p(\mathbf{m}_i, \mathbf{m}_{n_k^p(i)})}{d^p(\mathbf{m}_i, \mathbf{m}_{n_k^o(i)})} \right)^{1/2j}, \tag{7}$$

where $d^o(,)$ and $d^p(,)$ denote the distances in the original feature space and the projection space, and $n_k^o(i)$ and $n_k^p(i)$ denote the indices of the kth nearest neighbor of i in the original feature space and the projection space, respectively. Euclidean distances are used in both domains. In the map space, the distance between two nodes is the distance between the node coordinate vectors of the lattice. Values different from zero indicate mismatch between the original feature space and projection space; a negative value suggests a too low-dimensional map lattice and a positive value a too high-dimensional map.

Maps with lattice dimensions five and six gave values close to zero, see Figure 5. Also the four-dimensional map with larger value of neighborhood function gave a value close to zero. For all maps Φ became smaller as the neighborhood function was decreased (since the maps

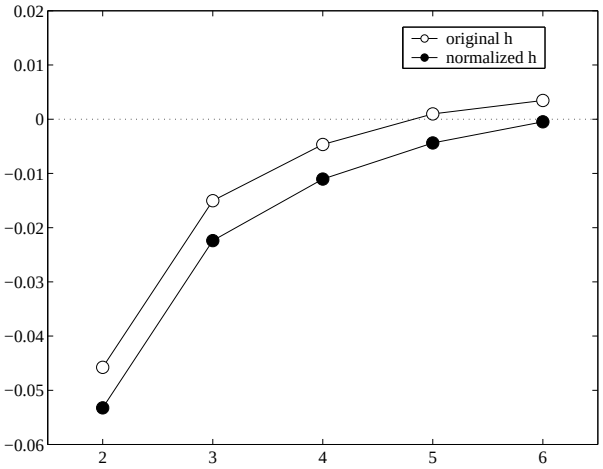

Figure 5. Topographic Product of hypercubical SOM lattices with dimensions 2 to 6.

became more flexible in the feature space). Although in theory the topographic product close to zero indicates a good match between the dimensions of the map lattice and feature space, in practice it is difficult to distinguish whether the map folding is a result from a too small lattice dimension or the true folding of the data manifold in the feature space.

2.3.4. *SOM training with continuous BMU trajectories*
The SOMs were also trained by forcing the BMUs of consecutive feature vectors to be nearest neighbors in the map lattice. This constraint was only applied in the BMU search. The updating of the prototypes was performed according to Equation 4. The quantization errors of the data sequences using these maps were very close to those of the maps trained without the continuous-BMU-trajectory constraint (Figures 3 and 4). The reason for this may be the symmetric neighborhood function which was used. Although the trajectory of the constrained BMUs is forced to match better the structure of the map lattice, the symmetric neighborhood function blurs different speech trajectories.

3. Discussion and related work

Some related work is listed and briefly discussed in this section. Self-Organizing Maps with the node connections representing feature sequence trajectories have been earlier considered in [14] and [21]. Topology Representing Network (TRN) [17] is an algorithm for representing

the data set as a graph where nodes are prototypes of data based on vector quantization and the connections between the nodes represent topological relationships between the prototypes. Compared to the SOM, TRN does not have a predefined node structure. It has been used for finding the intrinsic dimension of the static data set by investigating the number of the node connections in the resulting graph [8]. However, at least with limited number of data samples (and codebook vectors) it seems that TRN based approach underestimates the true intrinsic dimension of the data [6]. Multi-layer perceptron (MLP) networks have also been used for finding the intrinsic dimension of data. MLPs can be used in an auto-associative mode converting the data to the latent variables and then trying to reconstruct the original data in the output layer [10]. The number of the units in the middle layer corresponds to the dimension of the hidden variable space and when the approach is successful, the intrinsic dimension of the data. An interesting approach to reveal the intrinsic dimension of speech and get a meaningful representation is to convert acoustic feature vectors to the movements of the articulatory system, i.e., to find an inverse mapping from the acoustic observations to the speech production system. Hidden Markov models with the topological structure of the states have been considered for this purpose in [19], see also [18]. Generative Topographic Mapping (GTM [5], a probabilistic generalization of the SOM) with transition probabilities between the hidden states has been investigated in [4].

4. Conclusions

The theme of this work was to find a low-dimensional representation space for speech data. A structured vector quantization codebook, the Self-Organizing Map, was used as a main tool. The SOM algorithm is usually applied to separate feature vectors without taking the temporal order of data into account. However, the temporal order of feature vectors is an important aspect of dynamic signals such as speech. The goal was to find a SOM lattice which is sufficiently low-dimensional, but which is able to represent the projections of the original feature vectors as continuous paths in the map lattice.

Fractal dimension of mel-scaled cepstral coefficient vectors computed from TIMIT sentences was determined using Grassberger-Procaccia's approach. This suggested the maximum dimension of six for the hypercubical SOM lattice. Multi-dimensional scaling based Curvilinear Component Analysis was also applied for the data but no clear conclusions could be made from these results. The representation error decreased smoothly as the dimension of the projection space increased.

Different SOM lattices were then experimented. The dimensions of the hypercubical SOMs varied between two and six. The quality of the maps was evaluated based on the quantization error and the continuity of the BMU trajectories in the map lattice. The quantization error of separate feature vectors was normalized by scaling the neighborhood function by the number of the nodes in the effective neighborhood, which depends on the dimension of the map lattice. When the feature vector sequences were fitted against the map lattices so that the ensuing BMUs were constrained to be contiguous map nodes, two-dimensional maps performed clearly worse compared to higher-dimensional maps. Differences between other maps were not as remarkable.

References

1. A. Aho, J. Hopcroft, and J. Ullman. *Data structures and algorithms*. Addison-Wesley, 1983.
2. H. Bauer and K. Pawelzik. Quantifying the neighborhood preservation of self-organizing feature maps. *IEEE Transactions on Neural Networks*, 3(4):570–579, 1992.
3. H. Bauer and T. Villmann. Growing a hypercubical output space in a self-organizing feature map. *IEEE Transactions on Neural Networks*, 8(2):218–226, 1997.
4. C. Bishop, G. Hinton, and I. Strachan. GTM through time. Technical Report 005, Neural Computing Research Group, Aston University, Birmingham, UK, 1997.
5. C. Bishop, M. Svensén, and C. Williams. GTM: the generative topographic mapping. *Neural Computation*, 10(1):215–234, 1998.
6. F. Camastra and A. Vinciarelli. Intrinsic dimension estimation of data: an approach based on Grassberger-Procaccia's algorithm. *Neural Processing Letters*, 14(1):27–34, 2001.
7. P. Demartines and J. Hérault. Curvilinear component analysis: A self-organizing neural network for nonlinear mapping of data sets. *IEEE Transactions on Neural Networks*, 8(1):148–154, 1997.
8. F. Frisone, F. Firenze, and P. Morasso. Application of topology-representing networks to the estimation of the intrinsic dimensionality of data. In *Proceedings of ICANN'95*, pages 323–327, Paris, France, 1995.
9. P. Grassberger and I. Procaccia. Measuring the strangeness of strange attractors. *Physica*, 9D:189–208, 1983.
10. R. Hecht-Nielsen. Replicator neural networks for universal optimal source coding. *Science*, 269(5232):1860–1863, 1995.
11. S. Kaski and K. Lagus. Comparing self-organizing maps. In *Proceedings of ICANN'96*, pages 809–814, Bochum, Germany, 1996.
12. T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69, 1982.
13. T. Kohonen. *Self-Organizing Maps*. Springer, Berlin (3rd extended ed. 2001), 1995.

14. M. Kokkonen. Koartikulaatioilmiöiden mallittaminen itseorganisoituvan piirrekartan topologian avulla. Master's thesis, Helsinki University of Technology, Finland, 1991. In Finnish.
15. J. Lee, A. Lendasse, M. Verleysen. Curvilinear Distance Analysis versus Isomap. In *Proceedings of ESANN'2002*, pages 185–192, 2002.
16. Y. Linde, A. Buzo, and R. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28(1):84–95, 1980.
17. T. Martinetz and K. Schulten. Topology representing networks. *Neural Networks*, 7(3):507–522, 1994.
18. D. Nix and J. Hogden. Maximum-likelihood continuity mapping (MALCOM): An alternative to HMMs. In *Proceedings of NIPS'11*, pages 744–751, 1999.
19. S. Roweis. Constrained hidden Markov models. In *Proceedings of NIPS'12*, pages 782–788, 2000.
20. J. Sammon Jr. A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, 18(5):401–409, 1969.
21. P. Somervuo. Time topology for the self-organizing map. In *Proceedings of IJCNN'99*, volume 3, pages 1900–1905, Washington D.C., 1999.
22. TIMIT. DARPA TIMIT acoustic phonetic continuous speech database. CD-ROM, 1988.