Lunch talk, Tuesday 26 Februrary 2002, 12:45:06

Self-Organizing Maps for Signal and Symbol Sequences

Panu Somervuo

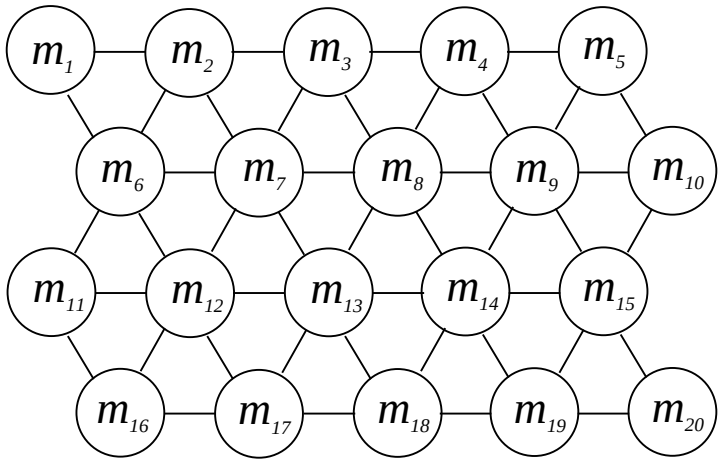
Neural Networks Research Centre
Helsinki University of Technology
Finland
http://www.cis.hut.fi

1. Self-Organizing Map for vectorial and nonvectorial data
2. Learning Vector Quantization
3. Examples

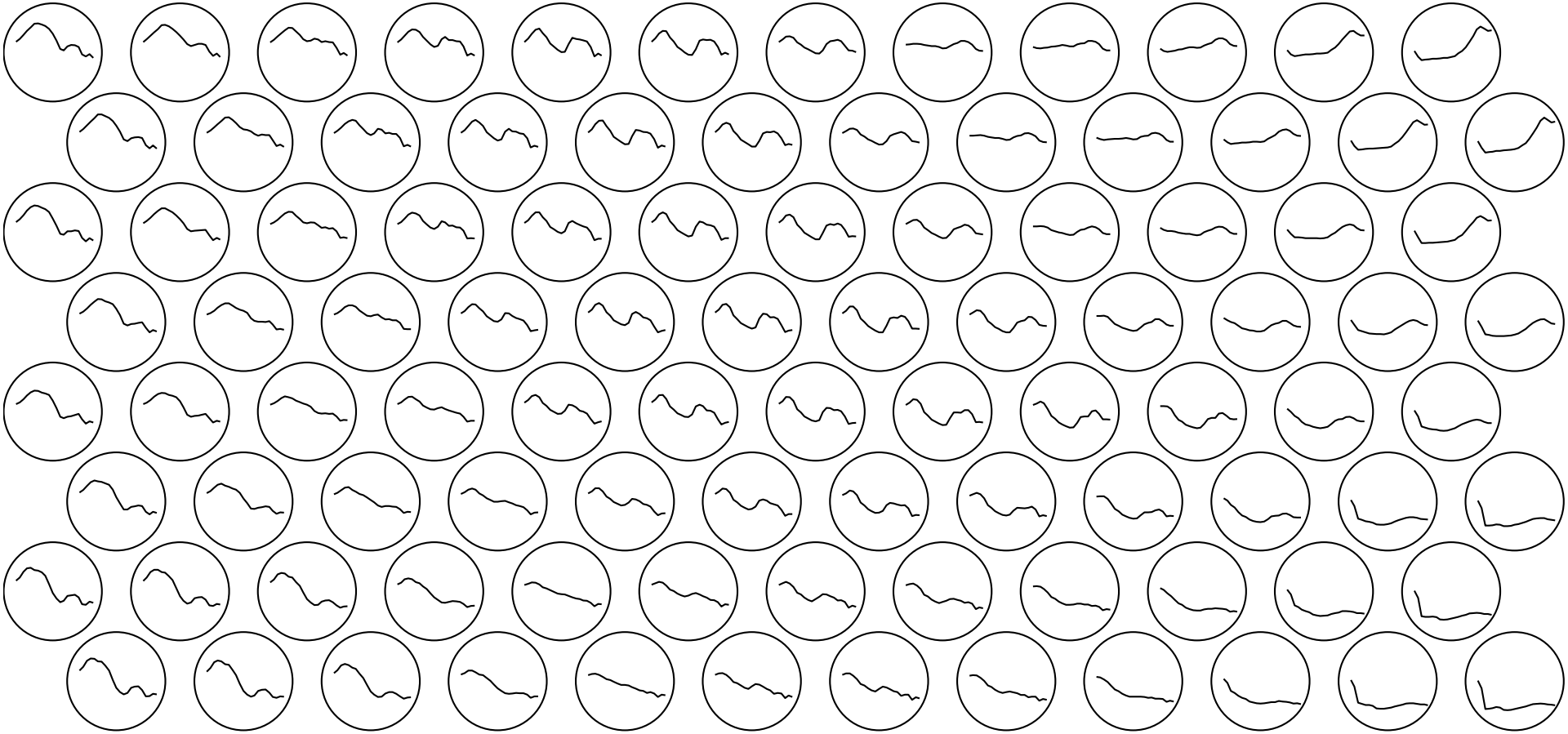
"A year in a laboratory can save you a day in a library"

Self-Organizing Map, SOM (Kohonen 1981)

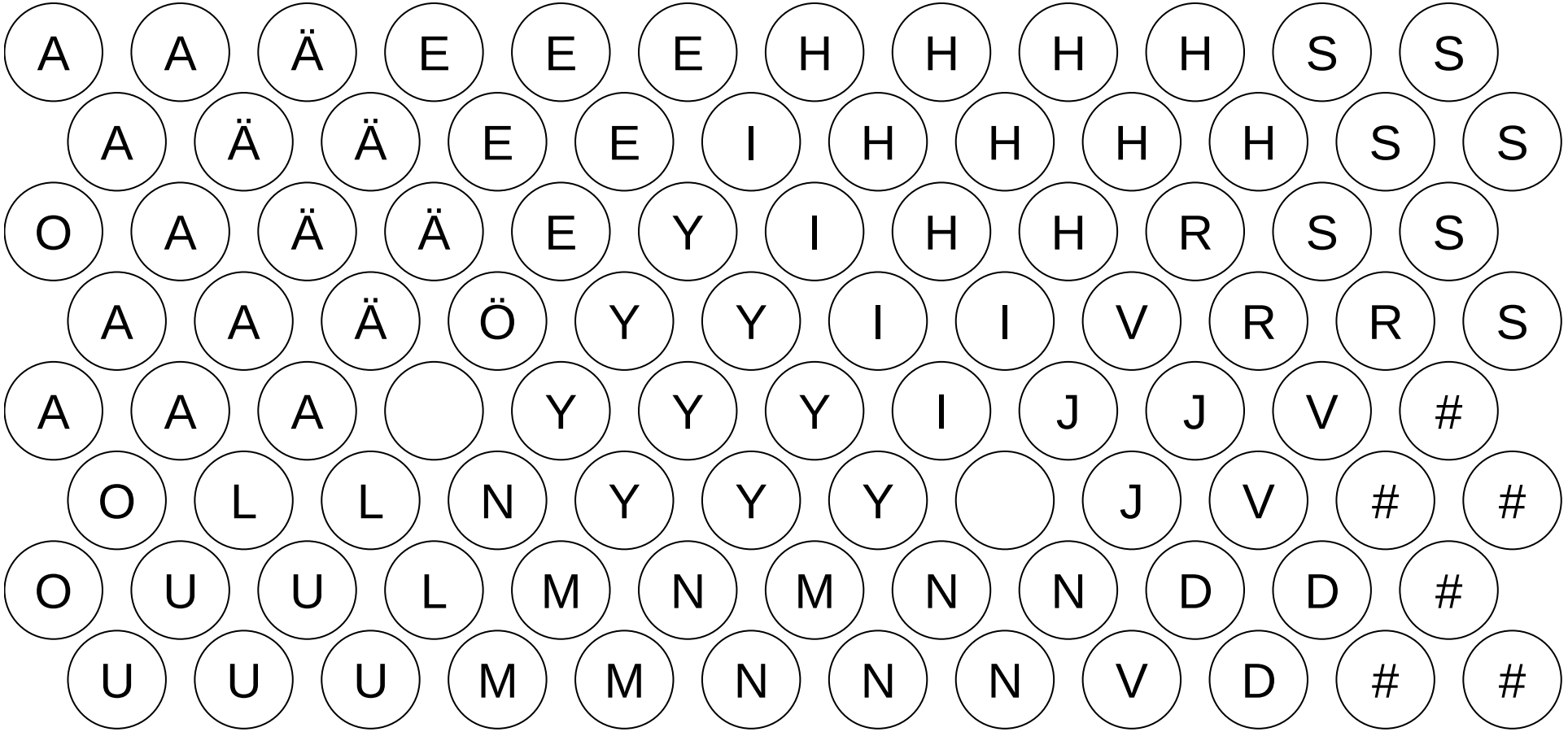
- "Artificial neural network", regular low-dimensional grid
- Each grid node is associated with the model m_i of the data
- Map learns representation of the data through an **unsupervised** learning process
- Learning principle: use **competitive learning** and **neighborhood** when adapting the models. Repeat:
  1. find the best-matching unit (BMU) for the input
  2. update the model of the BMU and the models belonging to the neighborhood of this unit
- Random initialization $\rightarrow$ ordering



Feature vector prototypes on the (trained) SOM



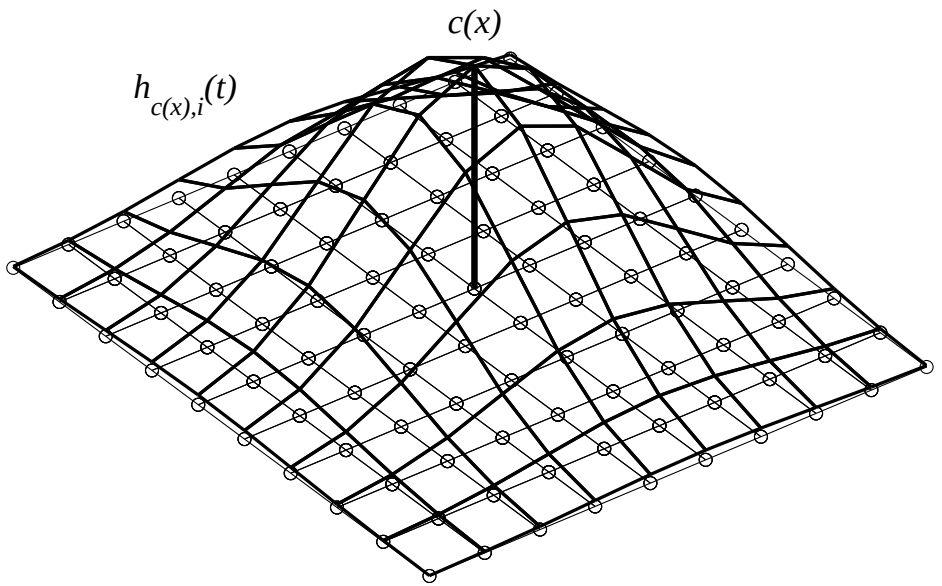
SOM with phoneme labels



SOM algorithm for vectors, online-learning

- Initialize the models (e.g. randomly)
- Repeat the following two steps for the samples $\mathbf{x}$ taken from the training set:
  1. Find the best-matching unit:
  $$c(\mathbf{x}) = \arg\min_i \{||\mathbf{x} - \mathbf{m}_i||\}$$
  2. Update the models:
  $$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + h_{c(\mathbf{x}),i}(t)[\mathbf{x} - \mathbf{m}_i(t)]$$

The neighborhood function $h_{c(\mathbf{x}),i}(t)$ usually decreases during the training.



SOM algorithm for vectors, batch-learning

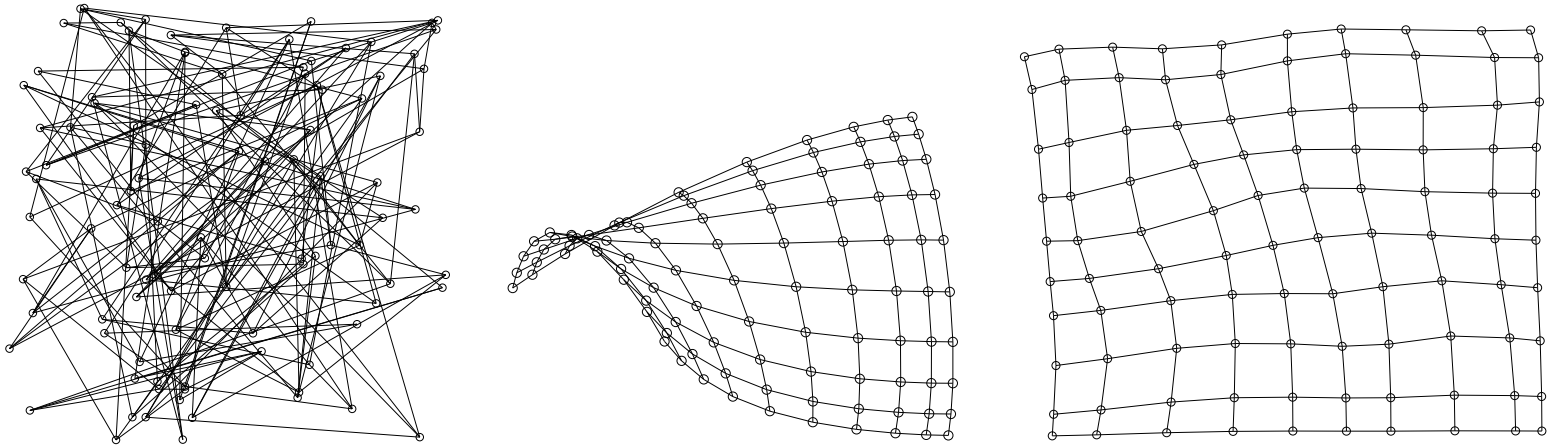
- Initialize the models
- Repeat steps 1 and 2 until convergence:
  1. First find the best-matching units for **all** input samples $\mathbf{x}_j$:

$$c(\mathbf{x}_j) = \arg\min_i \{||\mathbf{x}_j - \mathbf{m}_i||\}$$

  2. Then update the models:

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

Example of ordering from random initialization: data set is 2-dimensional unit square

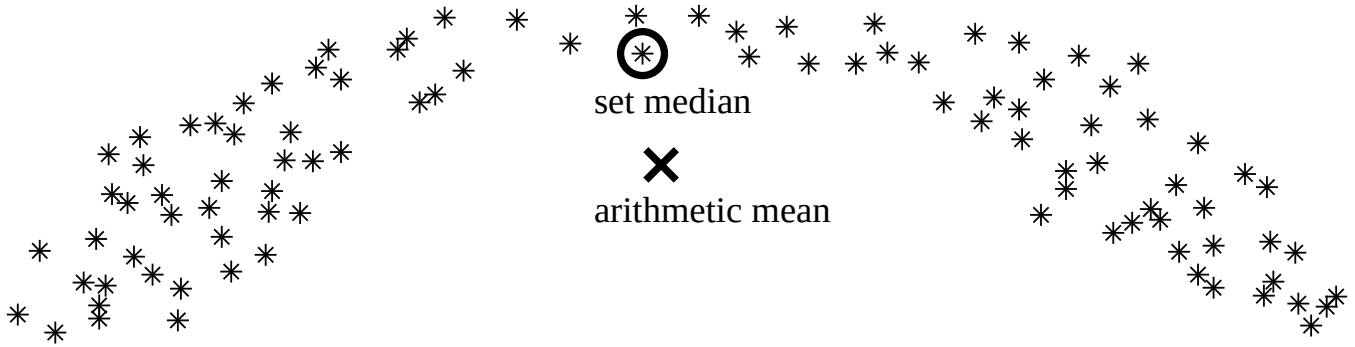


Generalizing the SOM for nonvectorial data

- We need:
  1. Distance or similarity measure between data items
  2. Model adaptation mechanism
- If we only have a distance measure between data items, how to update the models?
- Given a set of data items $\mathcal{S}$ and a distance measure d, the "centermost" data item m (median) satisfies the relation

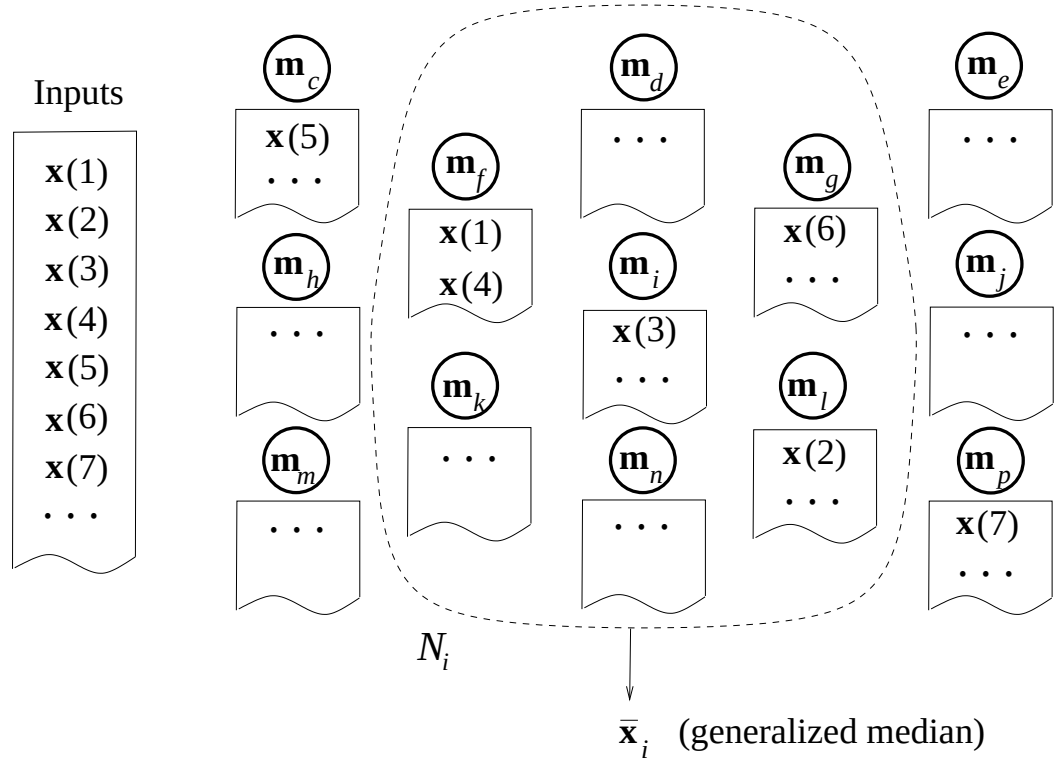
$$\sum_{x_j \in \mathcal{S}} d(x_j, m) = \min!$$

If m belongs to $\mathcal{S}$ it may be called the **set median**.



SOM algorithm for nonvectorial data

1. Initialize prototypes on the map. Random initialization may suffice, or alternatively, take samples that are ordered two-dimensionally (e.g. in the Sammon projection).
2. For each map unit i, collect a list of those data items to whom the prototype of unit i is the best representative.
3. For each map unit i, take for a new prototype the median over the union of the lists that belong to the topological neighborhood set N_i of unit i.
4. Repeat from 2 a sufficient number of times, until the prototypes are no longer changed in further iterations.



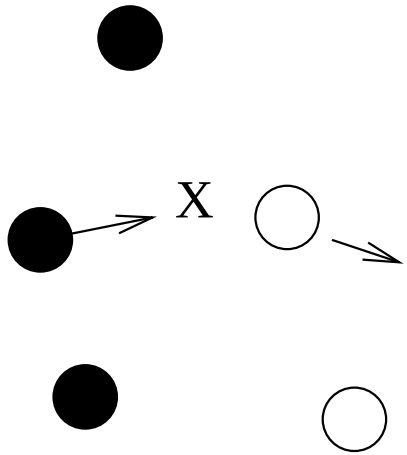
Learning Vector Quantization

SOM does unsupervised learning

LVQ is a supervised method:

$$\mathbf{m}_c(t+1) = \mathbf{m}_c(t) + \alpha(t)[\mathbf{x} - \mathbf{m}_c(t)]\,, \qquad \text{if } \mathbf{x} \in C(\mathbf{m}_c)$$

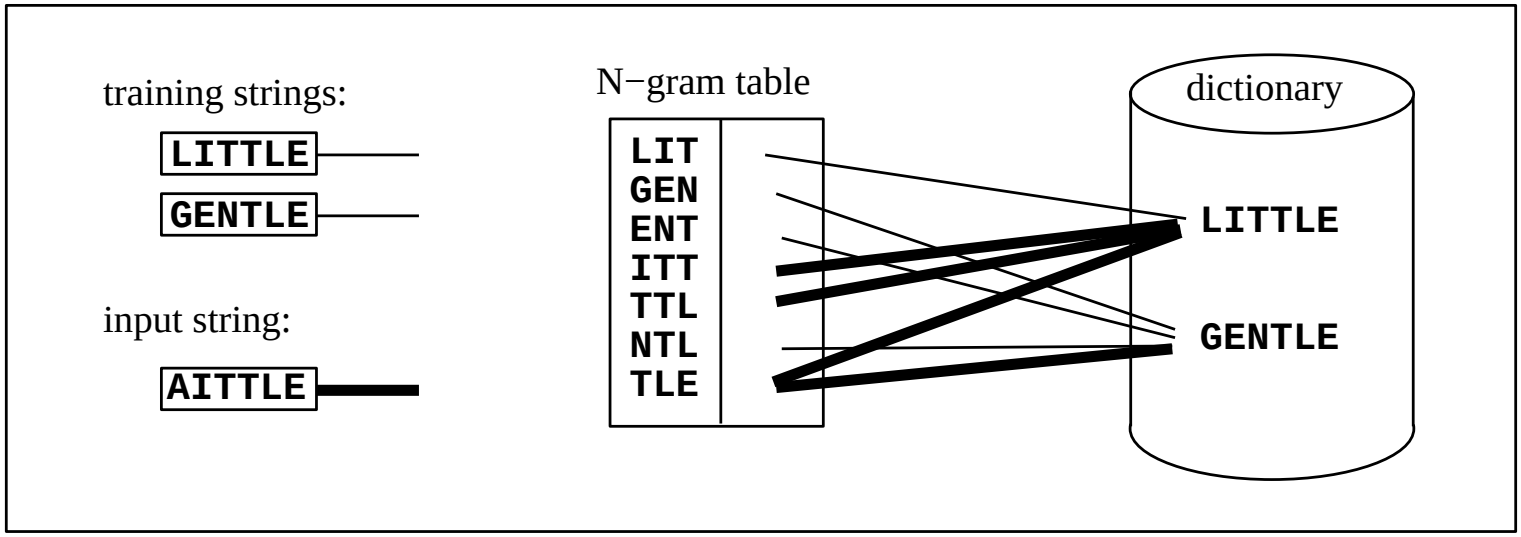
$$\mathbf{m}_c(t+1) = \mathbf{m}_c(t) - \alpha(t)[\mathbf{x} - \mathbf{m}_c(t)]\,, \qquad \text{else}$$



SOM and LVQ based prototypes for speech recognition

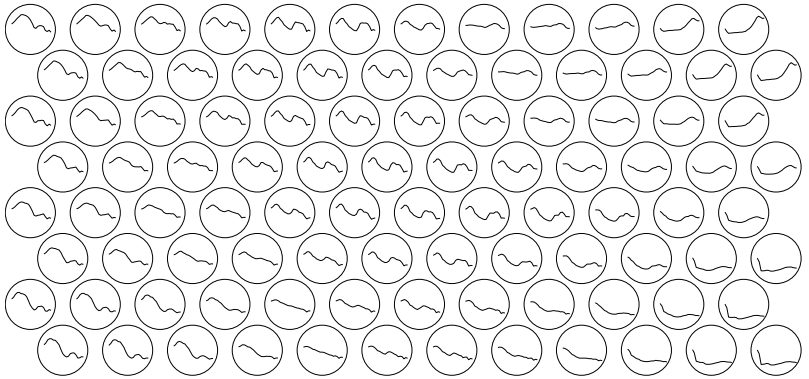
Model associated with each SOM node:	Application:
1. feature vector	VQ
	kernel means of mixture-of-Gaussian models
2. feature vector sequence	variable-length word models
3. hidden Markov model	set of (non-linguistic) speech segments
4. symbol string	learning pronunciation dictionary

Redundant Hash Addressing (Kohonen 1978)



$$FD(A,B) = \max(n_A, n_B) - n_d$$

RHA using the SOM nodes as alphabet

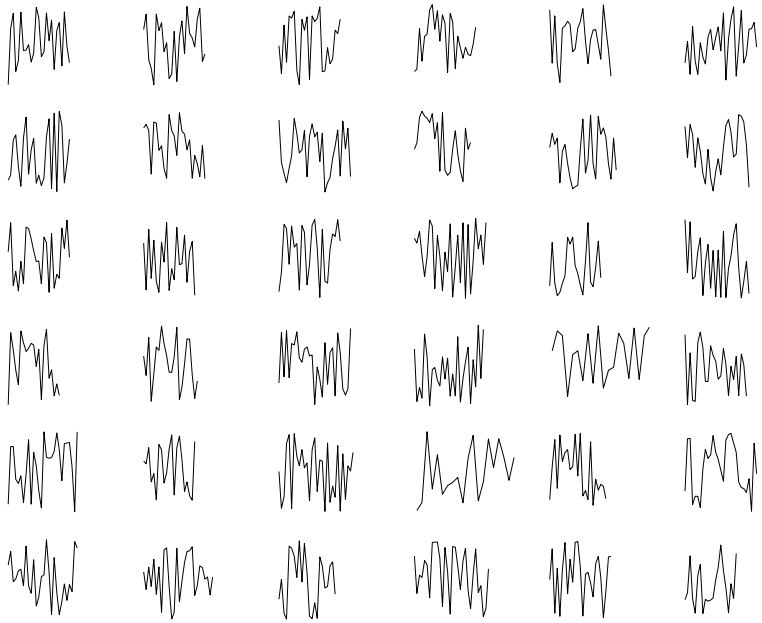


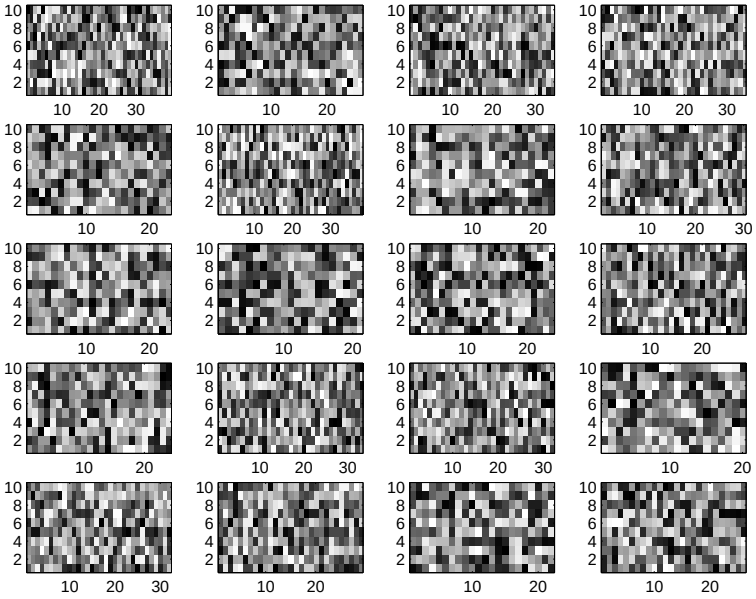
22 Finnish command words (20 speakers, 1760 utterances):

feature, recognition method	error %	time/ms
10-dim cepstrum, DTW	3.4	78
BMU index, Levenshtein	3.4	1060
BMU index, RHA N=2, d=150ms	6.6	5

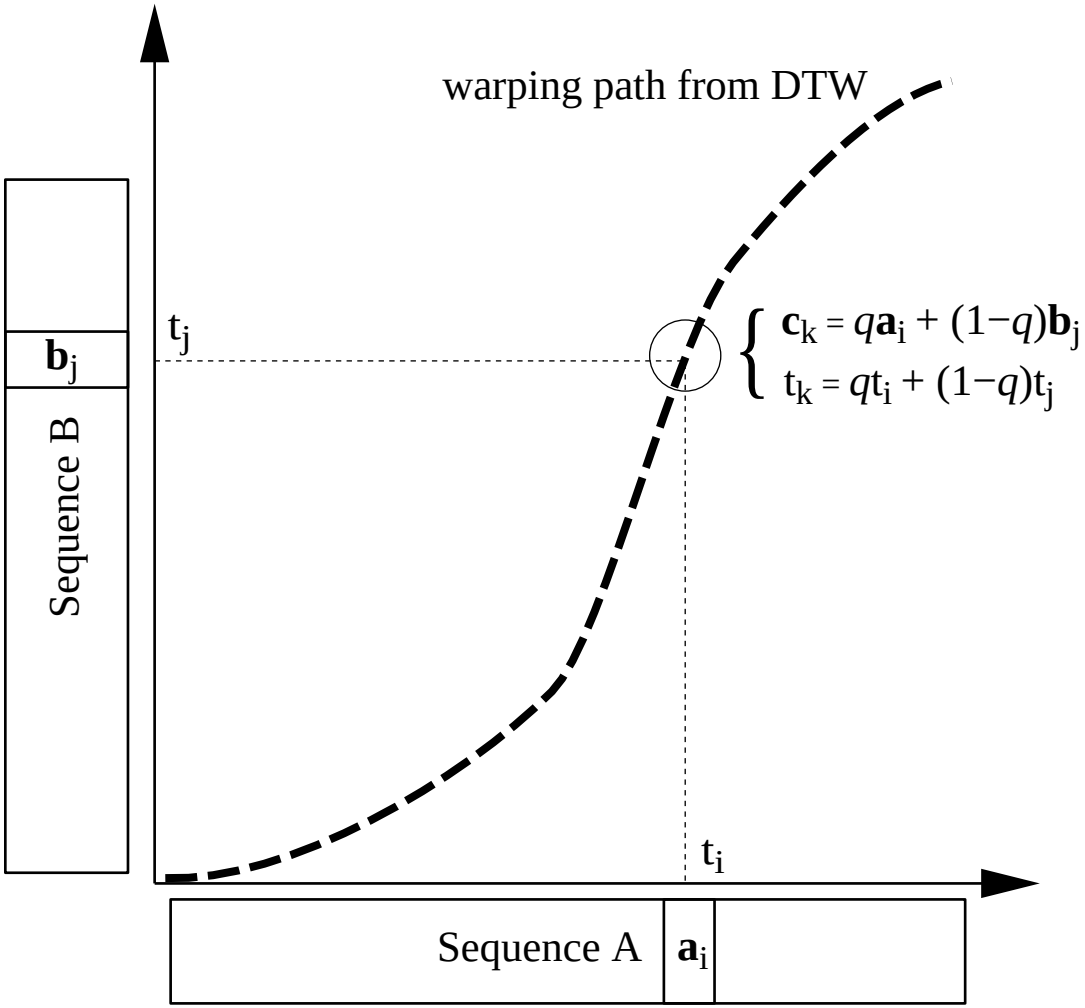
SOM with feature vector sequence prototypes

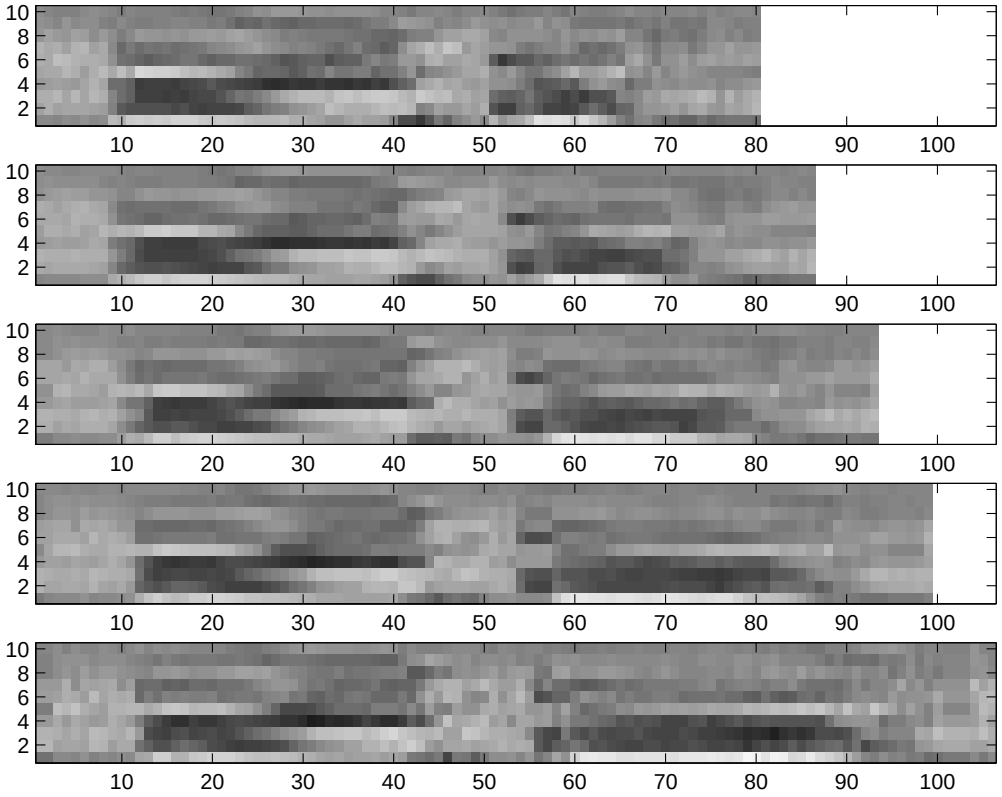
Initialization:





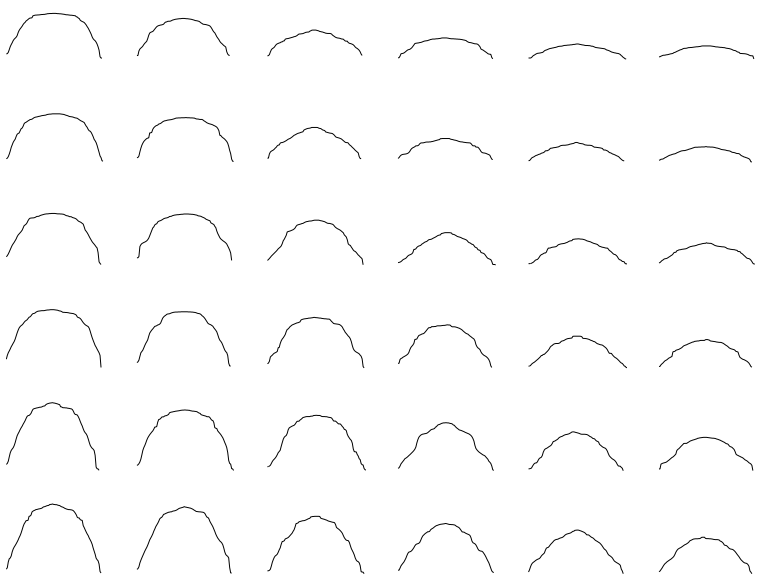
Averages of variable−length feature vector sequences

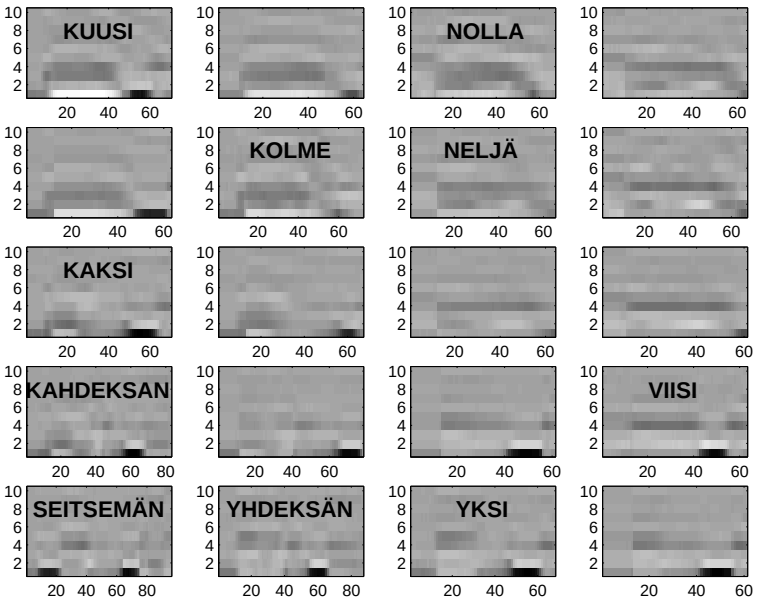




SOM for feature vector sequence prototypes

After training:

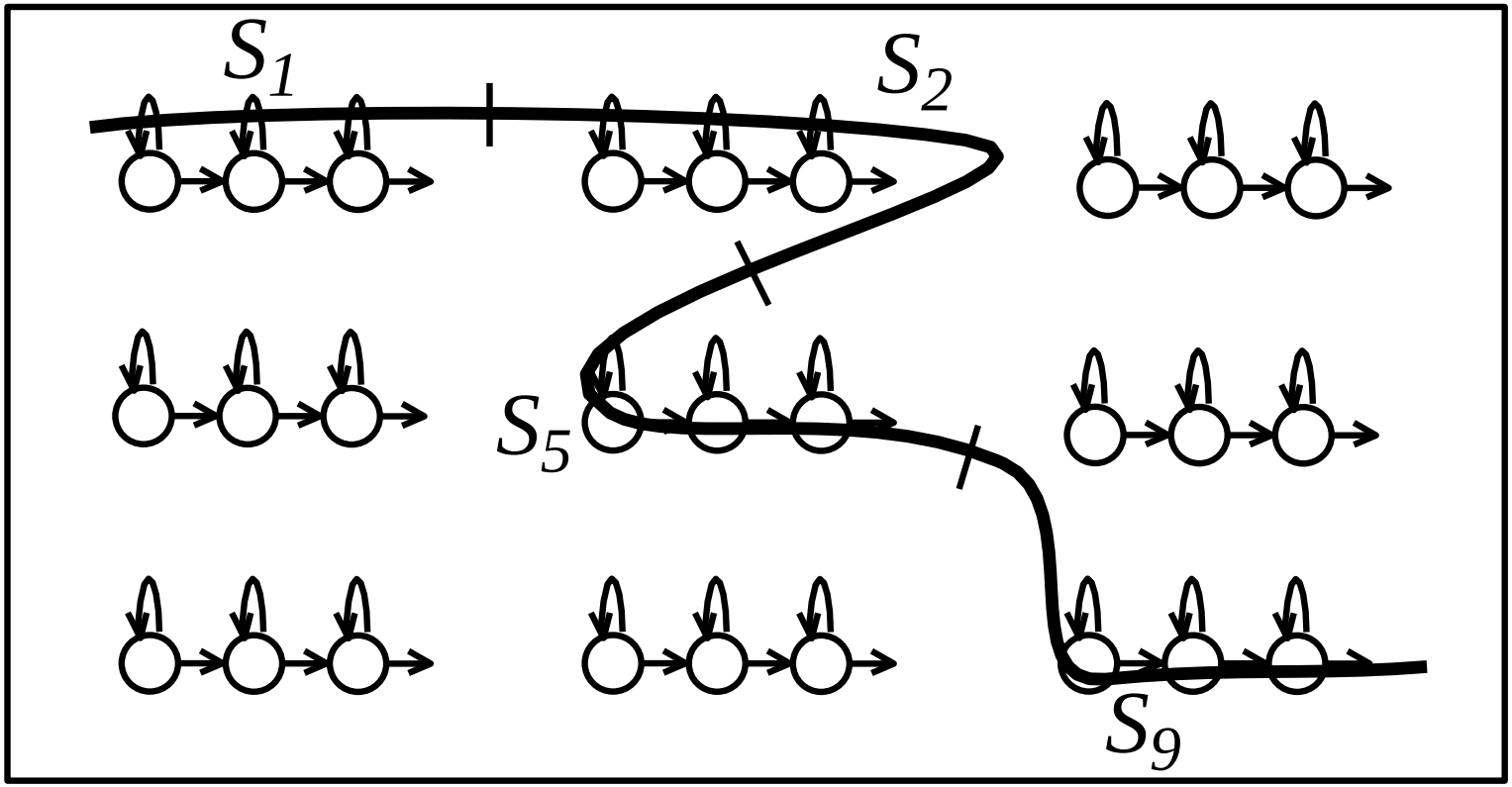




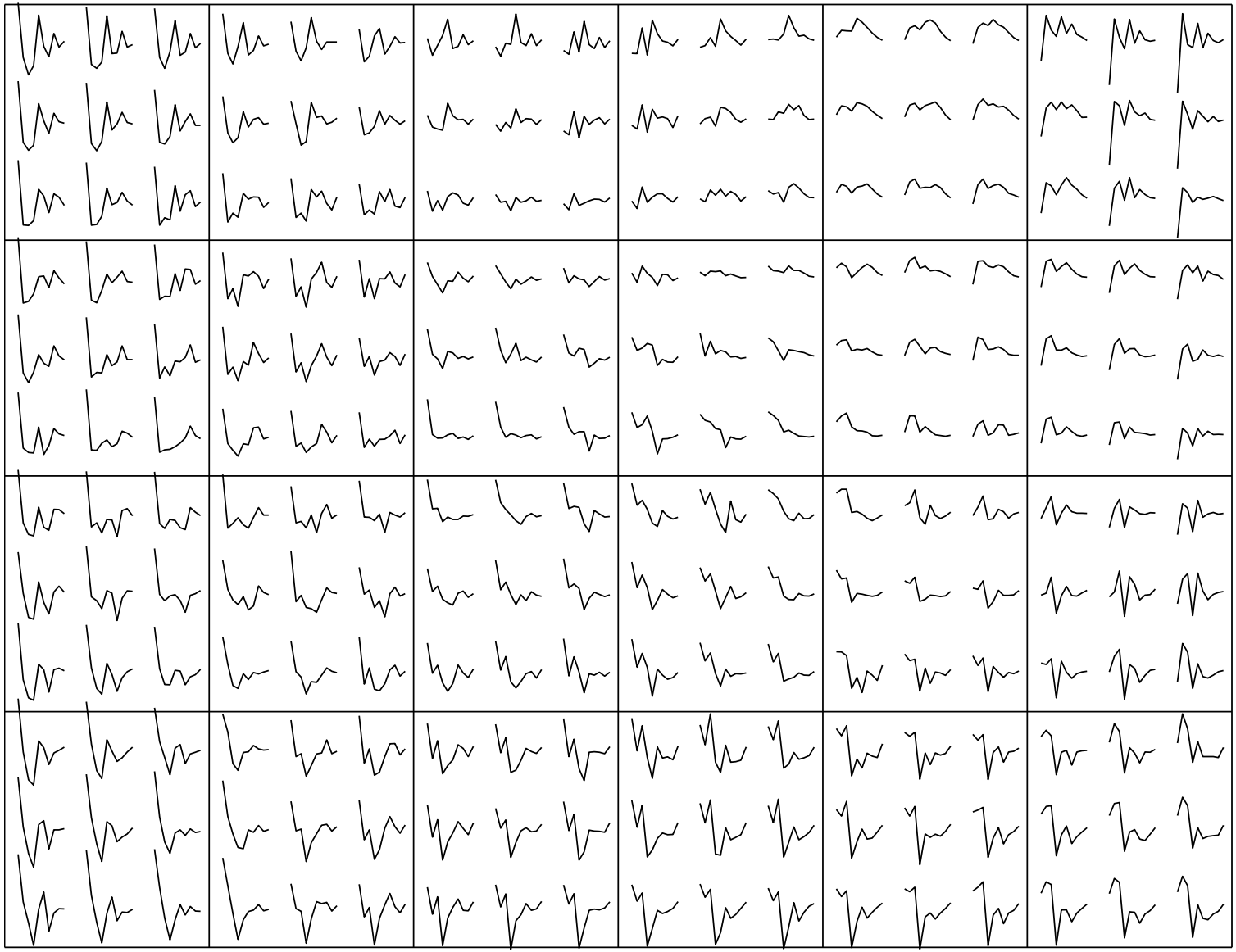
22 Finnish command words (20 speakers, 1760 utterances):

DTW, reference templates	error, per cent
one randomly picked sequence from each class	18.5
one centermost sequence from each class	3.1
one LVQ fine tuned sequence for each class	1.5

SOM with HMMs as nodes



6×4-node SOM, each node: single-state HMM modeled bythe mixture of 9 Gaussians:

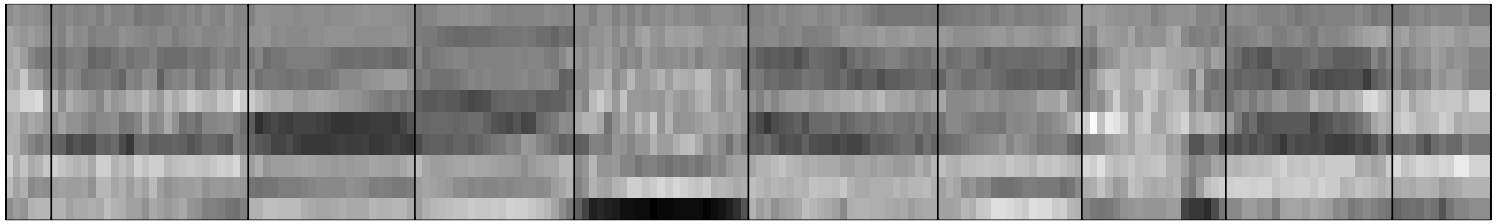


SOM with HMMs as nodes

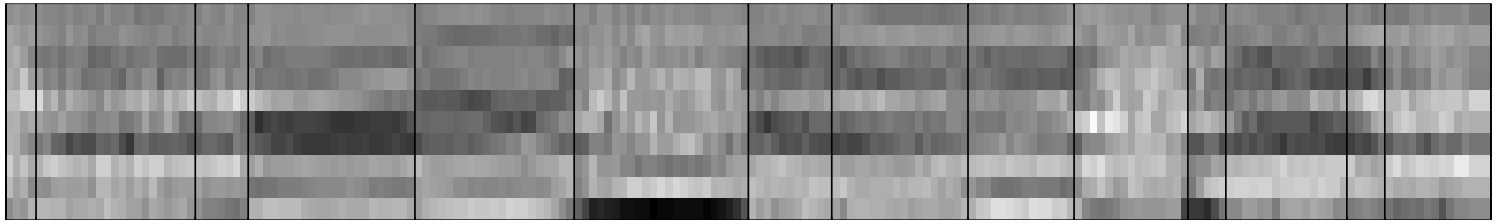
Trained SOM (unsupervised) for segmenting speech:

$/h/$ $/e/$ $/l/$ $/s/$ $/i/$ $/\eta/$ $/k/$ $/i/$

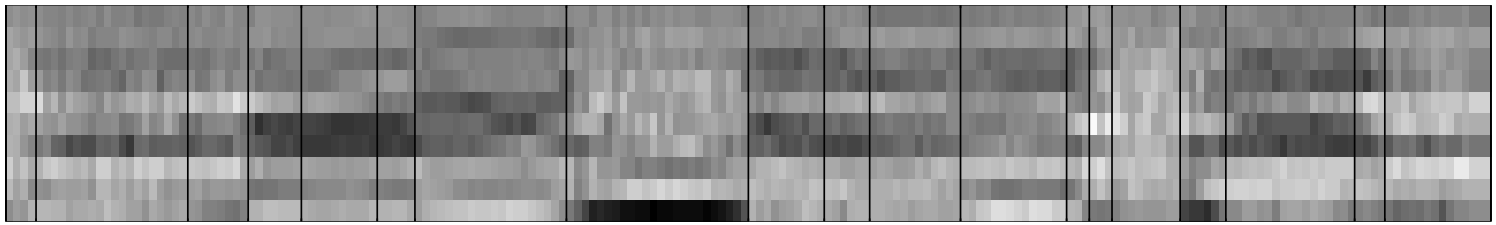
Phoneme borders:



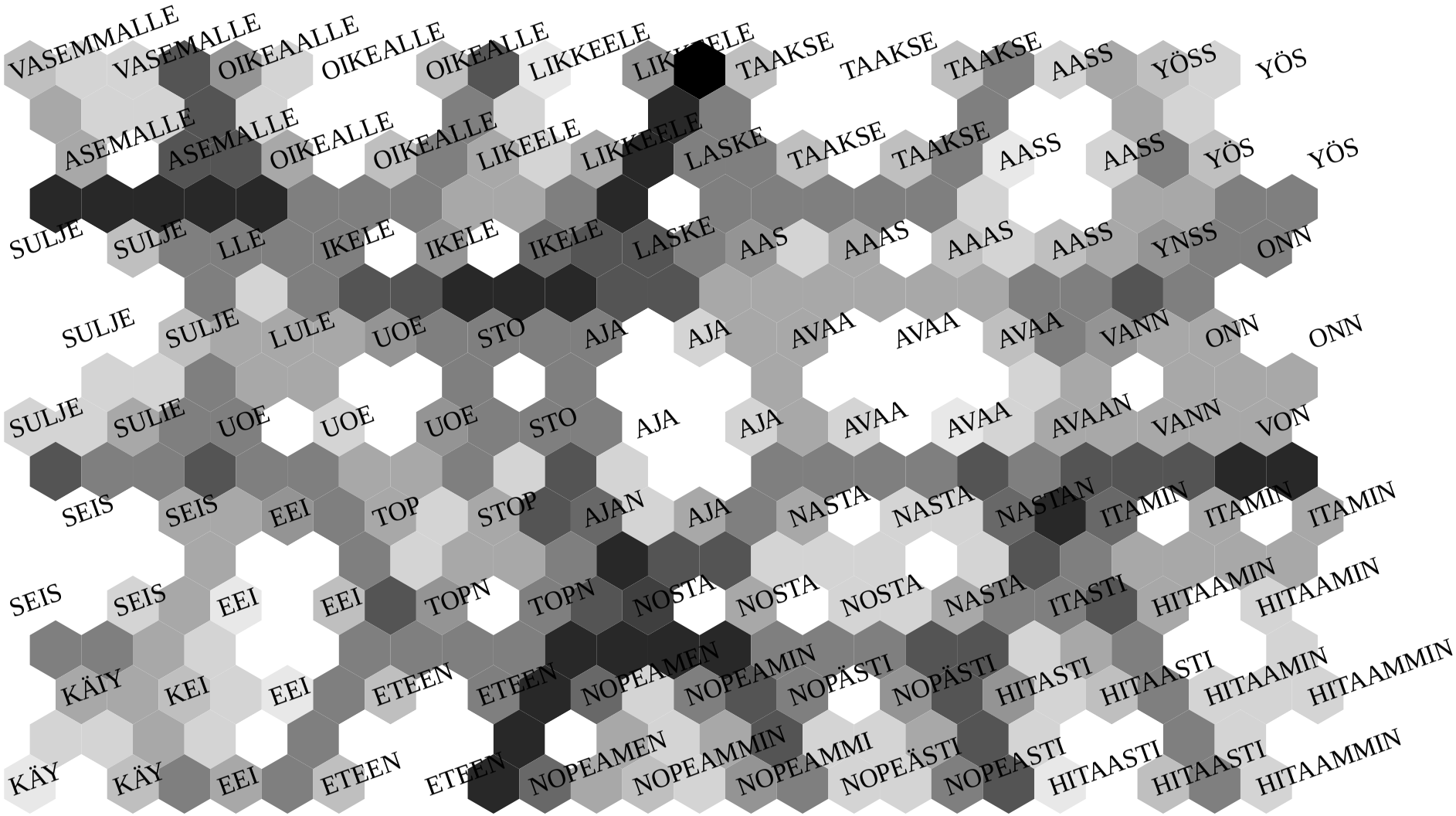
Segment borders using the SOM with 24 units:



Segment borders using the SOM with 216 units:



SOM of symbol strings



22 Finnish command words:

Mean strings

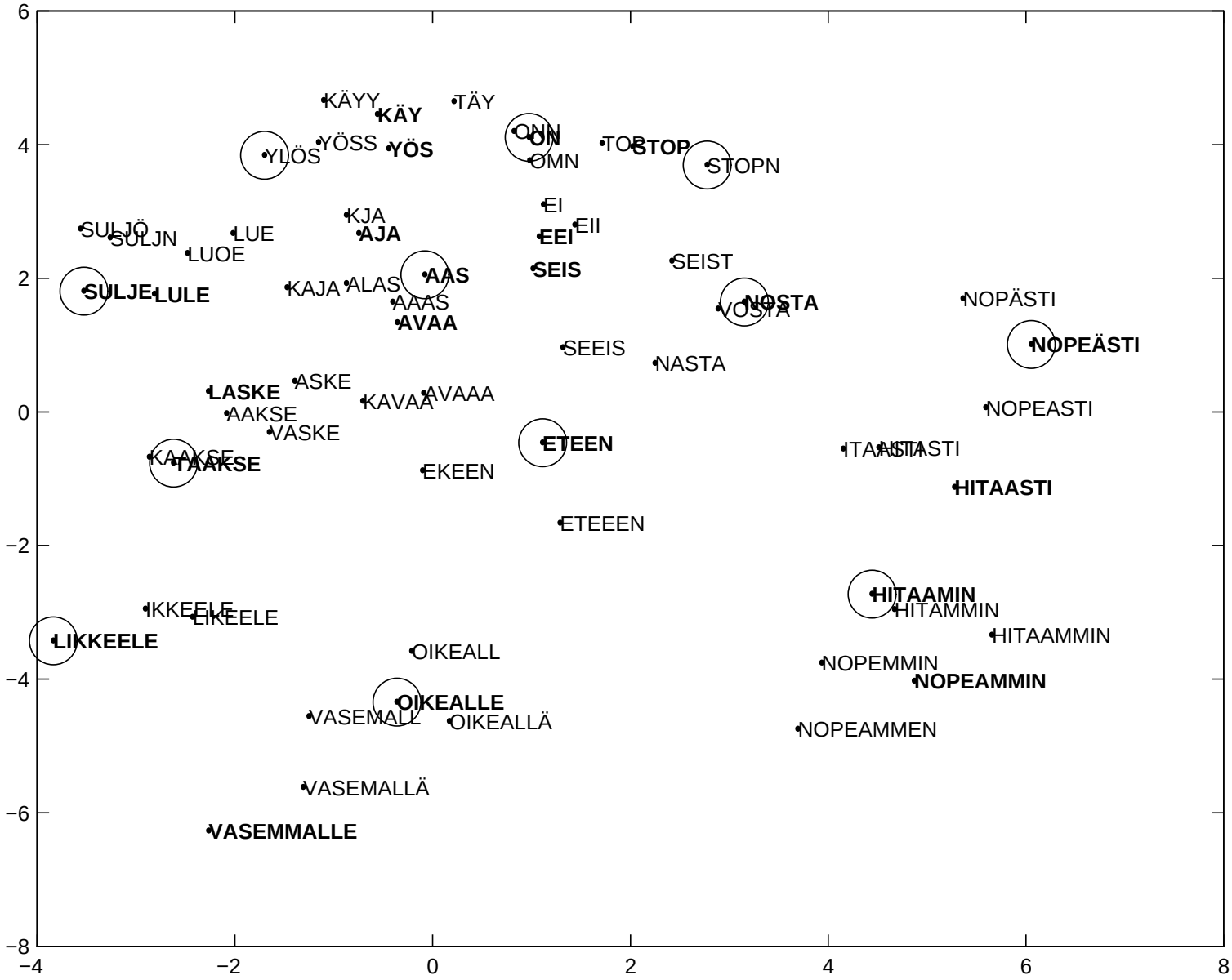
	training set	test set
SOM only, mean	4.1	4.4
SOM only, set mean	4.0	4.0
SOM + set LVQ1	3.5	3.8
SOM + LVQ1	2.6	2.7

Median strings

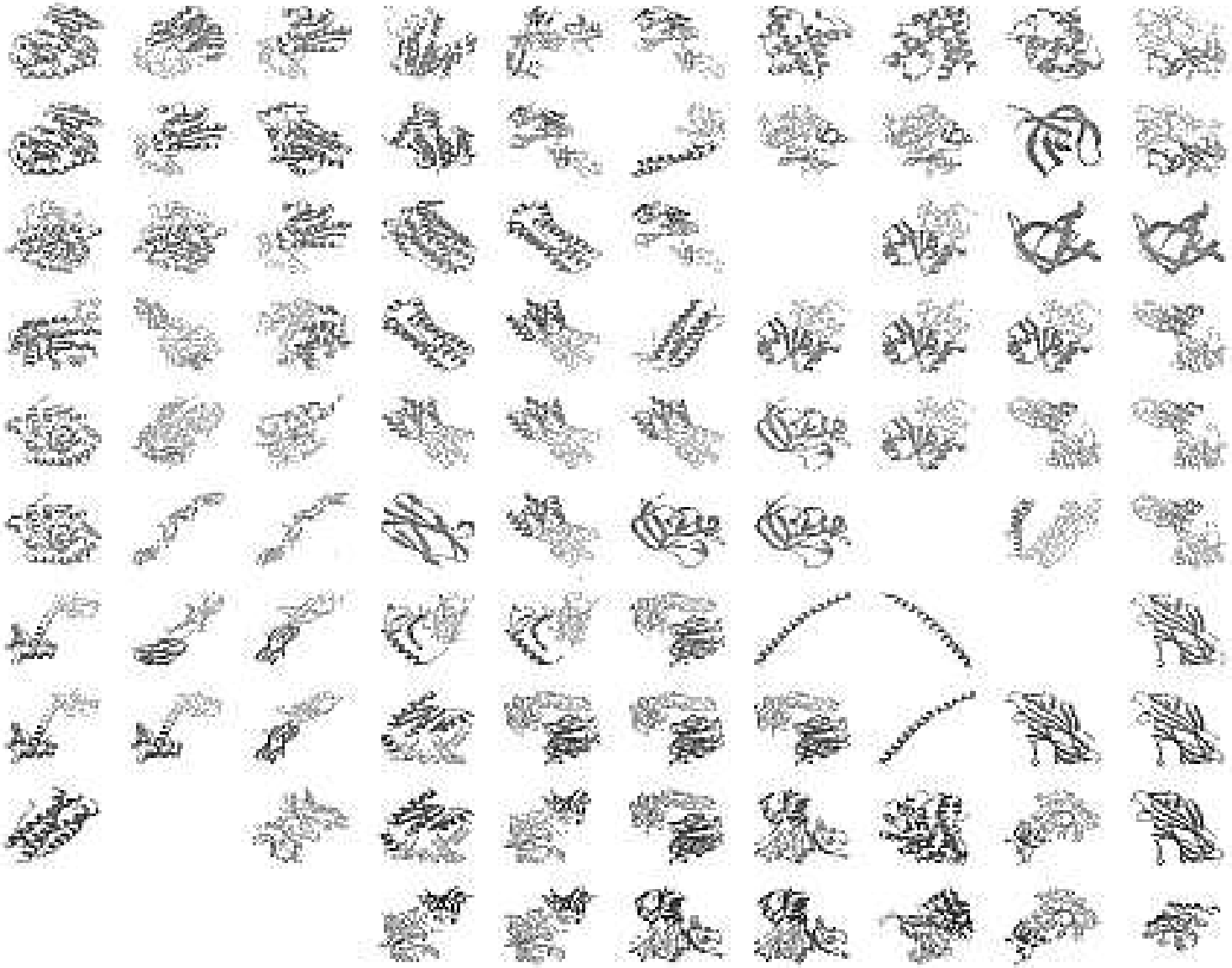
	training set	test set
SOM only, median	4.3	4.5
SOM only, set median	3.7	3.7
SOM + set LVQ1	3.4	3.5
SOM + LVQ1	3.2	3.3

Linguistic pronunciations: 5.9 % errors (LVQ considerably better!)

Sammon’s mapping (Sammon 1969)



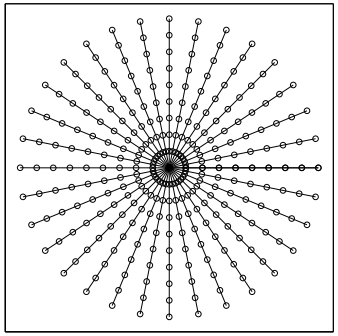
SOM of 3D protein molecules



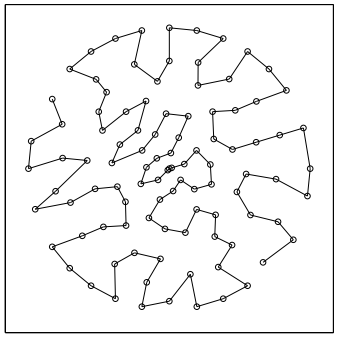
And now something completely different...

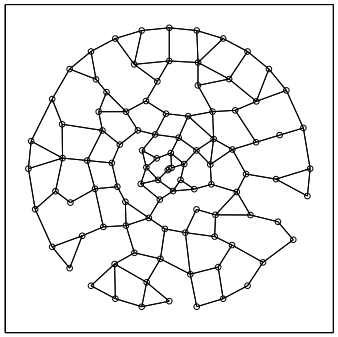
Data-driven node connections

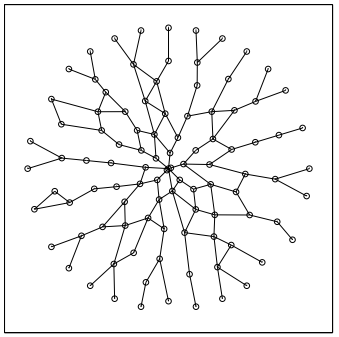
Data sequences:

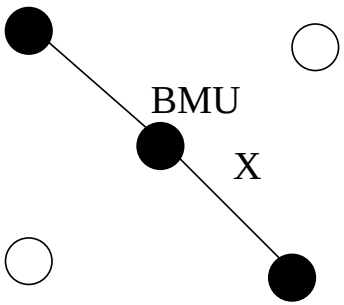


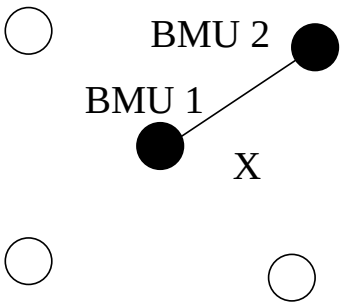
SOMs with different neighborhood connections:

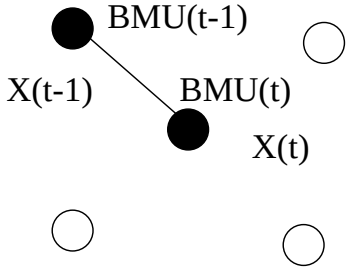


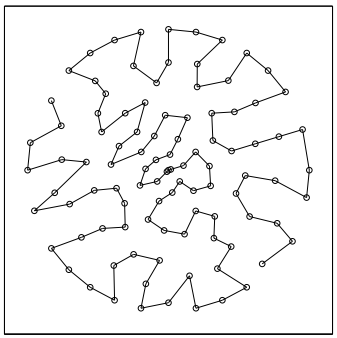


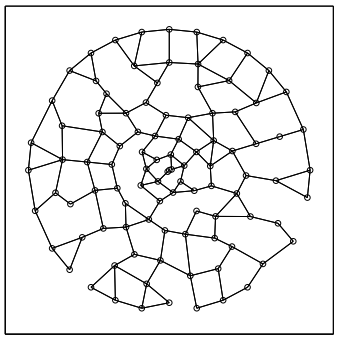


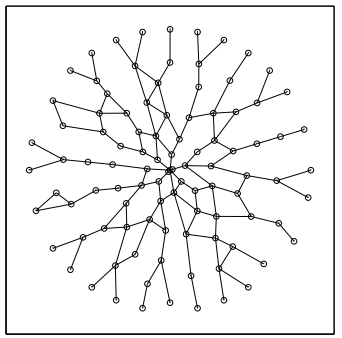

BMU
X


BMU 2
BMU 1
X


BMU(t-1)
X(t-1)
BMU(t)
X(t)





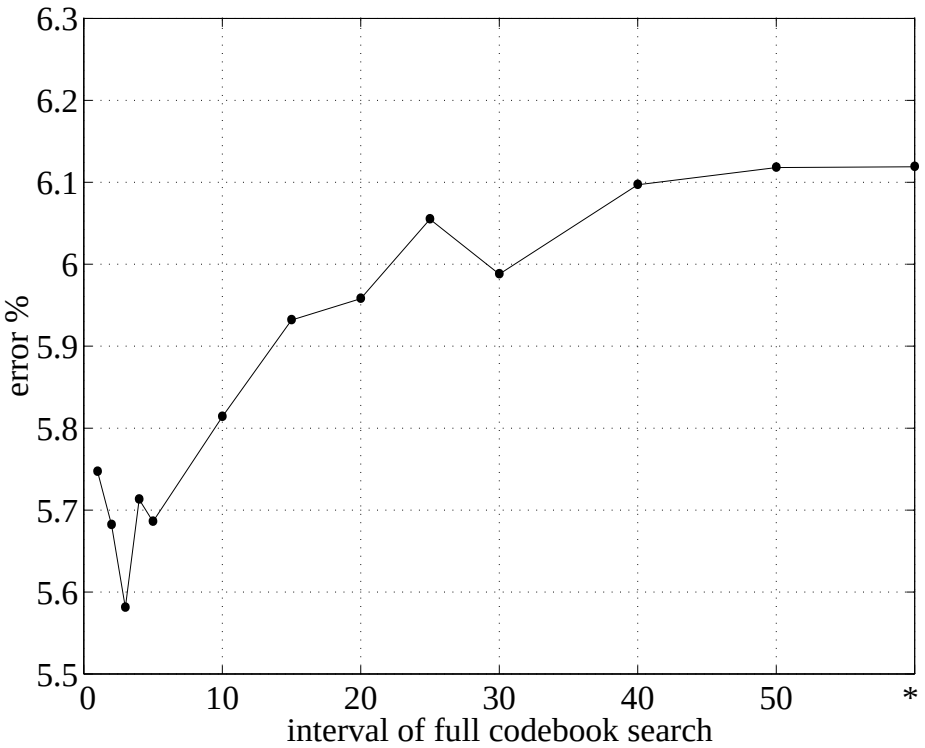


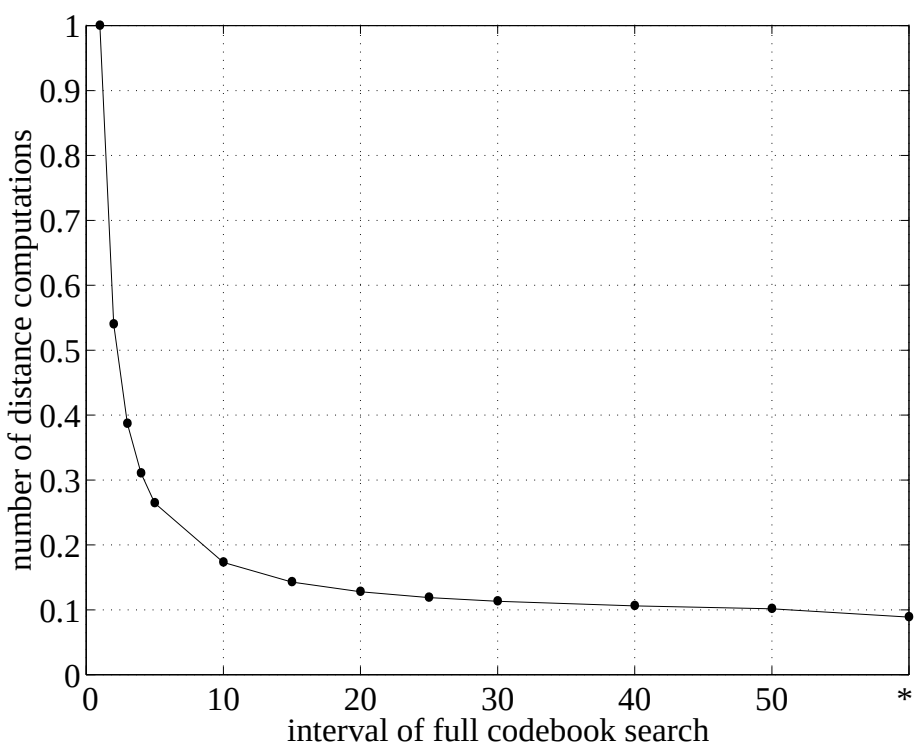
Application: speeding up HMM recognizer with mixture of Gaussians

Emission probabilities of the states are usually dominated by the nearest Gaussians to the input vector $\rightarrow$ try to find these Gaussians without performing full codebook search

- speech trajectories are modeled by means of the connections between the kernels
- during recognition, follow only the connections from the previous BMUs in each state

Task: speaker-dependent Finnish phoneme recognition (20 phonemes)
Data: 4×350 words from 10 speakers
Recognizer: 3-state left-right HMM for each phoneme each state modeled by 12×8-unit MOG (initialized by SOM)





Summary

- SOM and LVQ, heuristic algorithms, easy to implement
  - Unsupervised SOM tries to find topology preserving mapping
  - Supervised LVQ tries to give better class discrimination
- Usually SOM and LVQ applied for fixed-dimensional feature vectors. In this work:
  1. Predefined (regular) SOM grid, the model in each node:
     - feature vector (followed by RHA)
     - feature vector sequence
     - HMM
     - symbol string
  2. Data model is feature vector, but node connections are data-driven $\rightarrow$ "time-topology preservation"

(Unpublished work: Speech dimensionality analysis using hypercubical SOM grids)