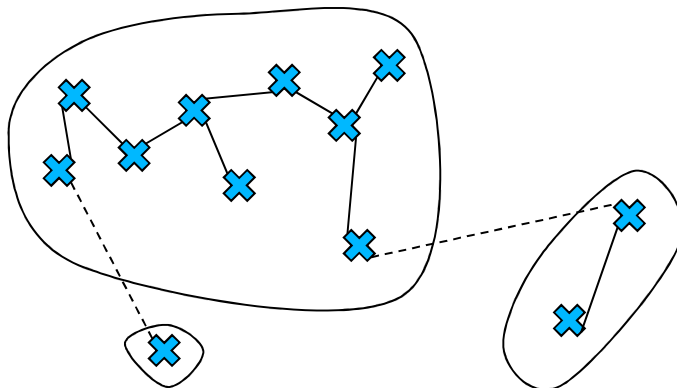


Example

Apply hierarchical clustering with d_{\min} to below data where $c=3$.

Nearest neighbor clustering



will form elongated clusters!

Computational Complexity of Hierarchical Clustering

Assume

n - samples

d - dimension

c - clusters to be formed.

Use $d_{\min}(D_i, D_j)$

- To find the nearest clusters at level \hat{c} ,
- $\theta(d)$ for each pair
- $n(n-1)$ pairwise calculations, $\theta(n^2)$ (each sample with all others except itself)
- Finding the minimum distance pair

overall $\propto \theta(cn^2d)$

NN clustering algorithm and minimal spanning trees

If we continue with NN to end up with a single cluster, we obtain "minimal spanning tree".

Spanning tree: a graph with no loops that will contain a path between any two points.

Minimal spanning tree: a spanning tree with a minimum total length (length : sum of path lengths)

Top-down Clustering: Graph theoretical approach so a top-down approach will

- Obtain the minimal spanning tree using a fast algorithm
- Remove the longest edge
- Continue removing until desired number of clusters are reached.

Solution with iterative clustering algorithm

Step 2 of Bottom-up hierarchical clustering algorithm:

Join clusters i and j if that will result with smallest increase in

$$\sqrt{\frac{n_i n_j}{n_i + n_j}} \|M_i - M_j\|^2$$

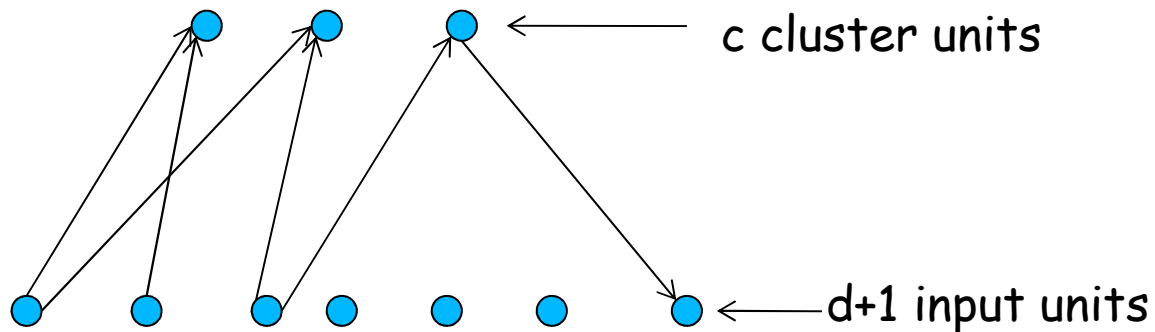
This is the same as joining clusters with closest means, normalized with the number of samples in each cluster.

One way use above to obtain an initial partitioning for iterative optimization!

$$d_{mean_{i,j}} = \|M_i - M_j\|$$

Competitive Learning and "Self-Organizing Feature Maps"

- A NN approach to clustering
- In all procedures that are satisfied, since a global criterion is minimized, an addition of a new sample changes the results dramatically.
- Competitive Learning: Only the clusters that are "close" to the new sample are affected.
- NN terminology used.
- Single layer NN

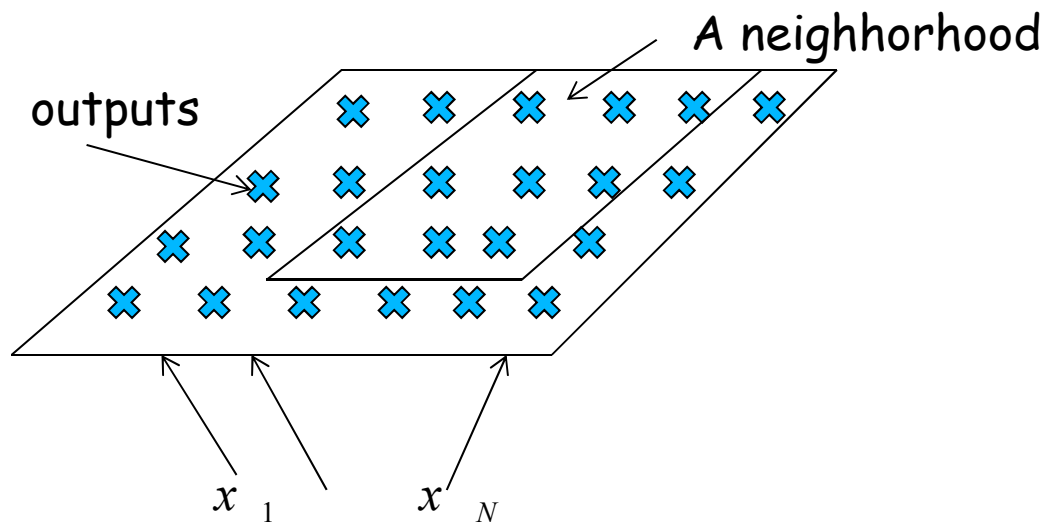


- Consider a fully connected network.

Kohonen's Self Organizing Feature Maps:

(Also called Vector Quantization)

- 2-d, single layer topology
- N inputs, k outputs
- Feed-forward, fully connected (all inputs connected to all outputs)
- Used for both supervised and unsupervised learning (clustering)
- Topologically similar inputs map to topologically close outputs
- Topological neighborhoods defined
- Neighborhoods start large and shrunk during learning.



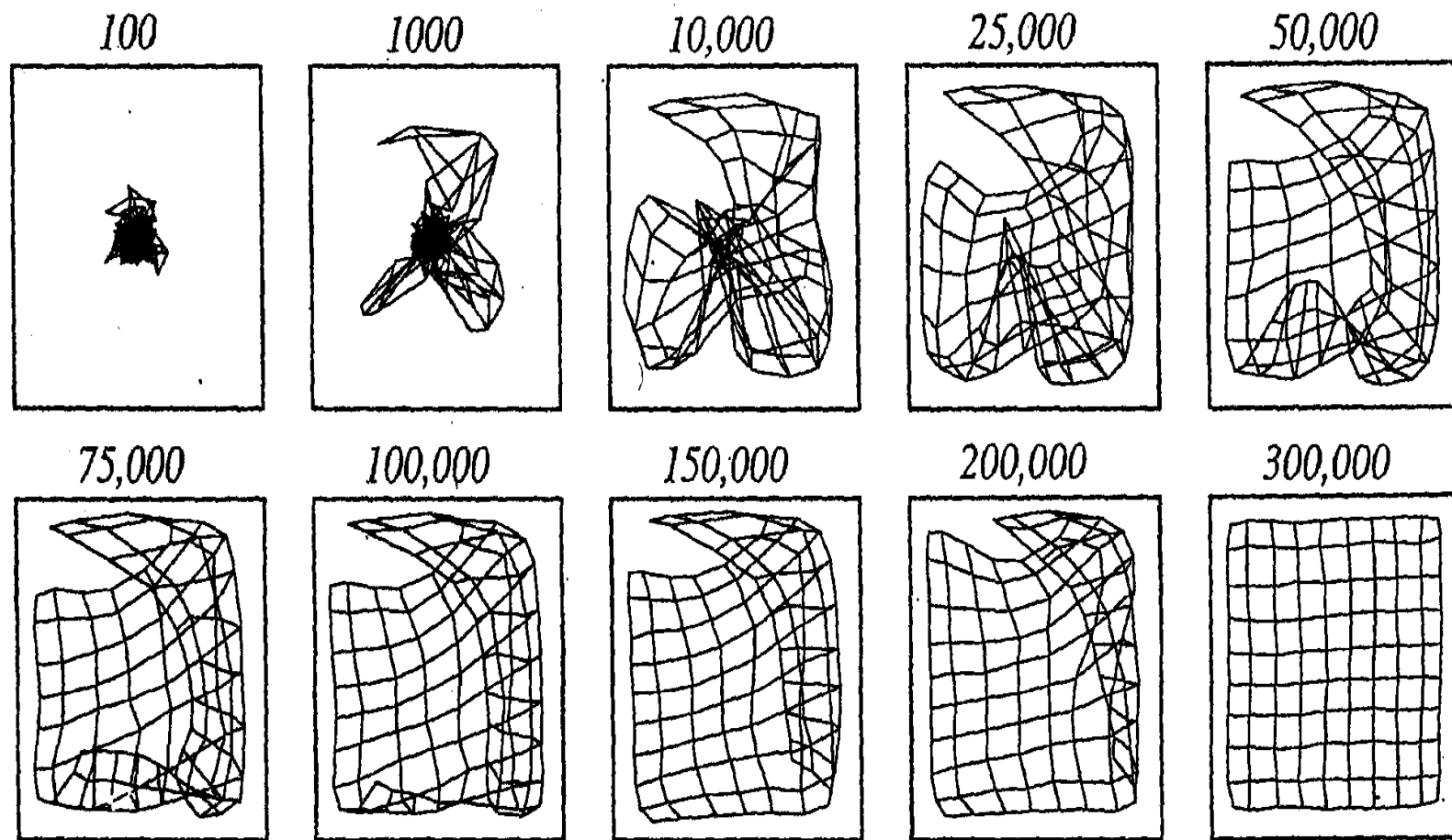


FIGURE 10.31. A self-organizing feature map from a square source space to a square (grid) target space. As in Fig. 10.28, each grid point of the target space is shown atop the point in the source space that leads maximally excites that target point.

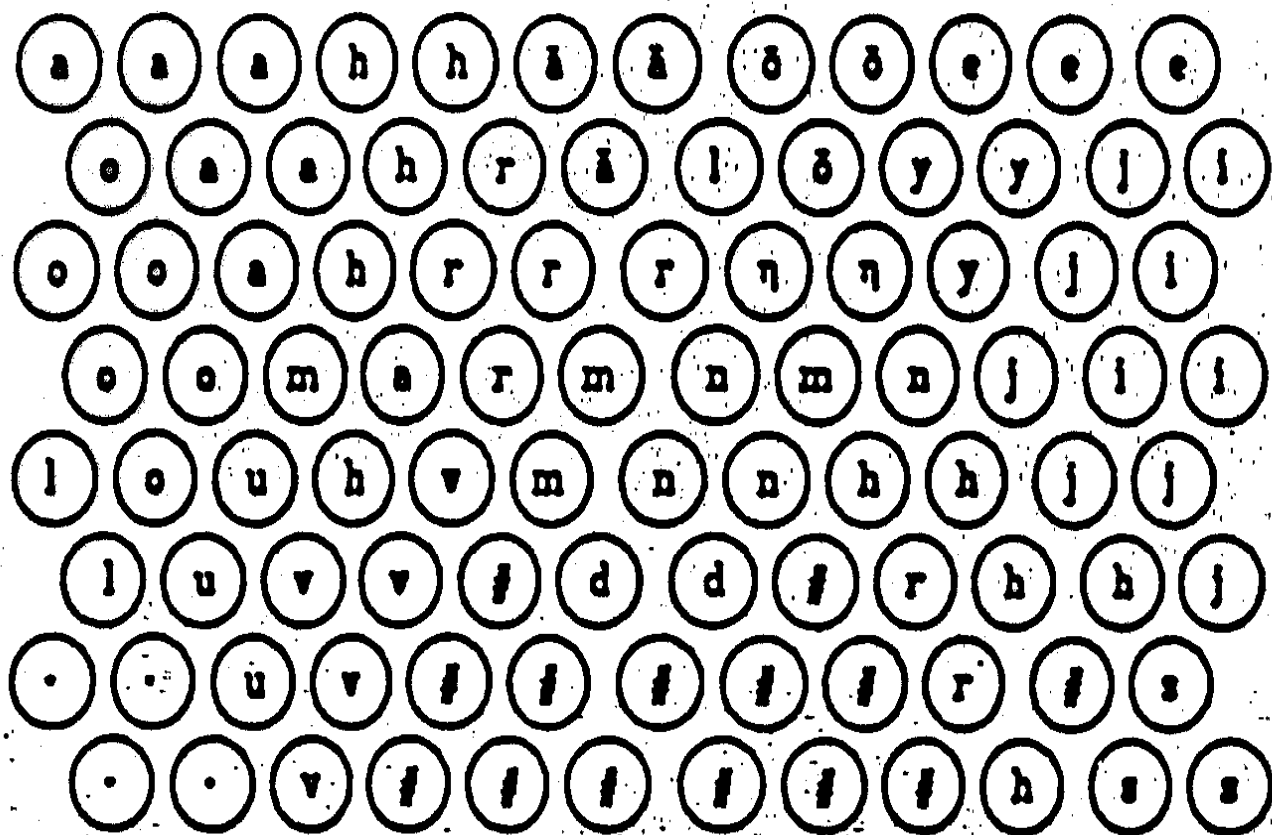


Fig. 3.16. The neurons, shown as circles, are labeled with the symbols of the phonemes with which they made the best match. Distinction of /k,p,t/ from this map is not reliable, and this phonemic class is indicated by symbol #. An analysis of the transient spectra of these phonemes by an auxiliary map is necessary

Basic Kohonen Learning:

- Initialize all weights w_{ij} to small random values.
- For all X , repeat.

1) Compute

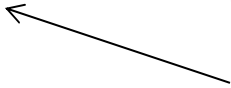
$$D(j) = \sum_{i=1}^N (w_{ij} - x_i)^2$$

2) Select node j^* such that $D(j)$ is minimum.

3) Update weights of nodes j^* and nodes in k in the neighborhood by $w_{ij}(new) = w_{ij}(old) + \sigma(x_i - w_{ij}(old))$

- Reduce σ . Reduce this neighborhood.
- Repeat until converges

A more general weight update :

$$w_{ij}(new) = w_{ij}(old) + \sigma(t)\Lambda(x_i - w_{ij})$$


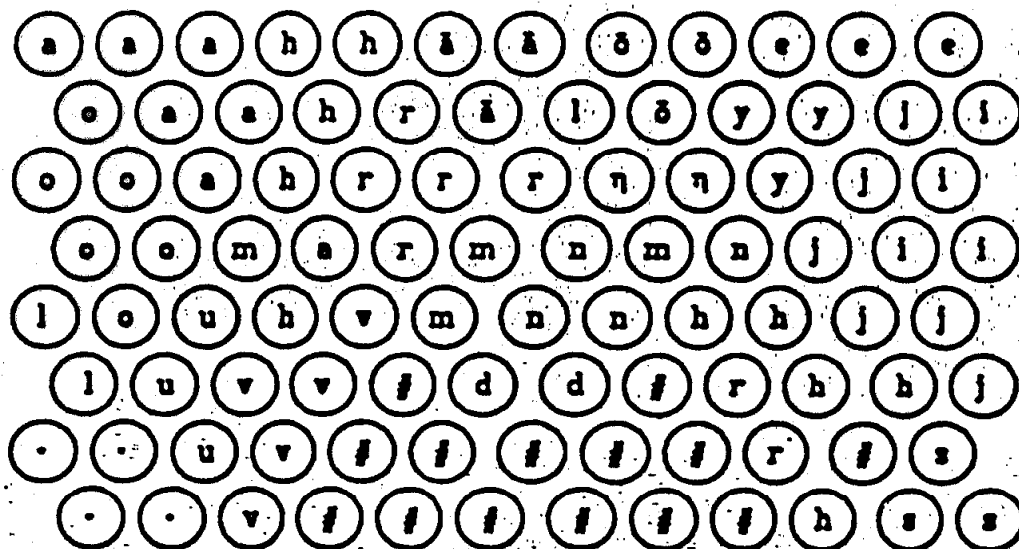
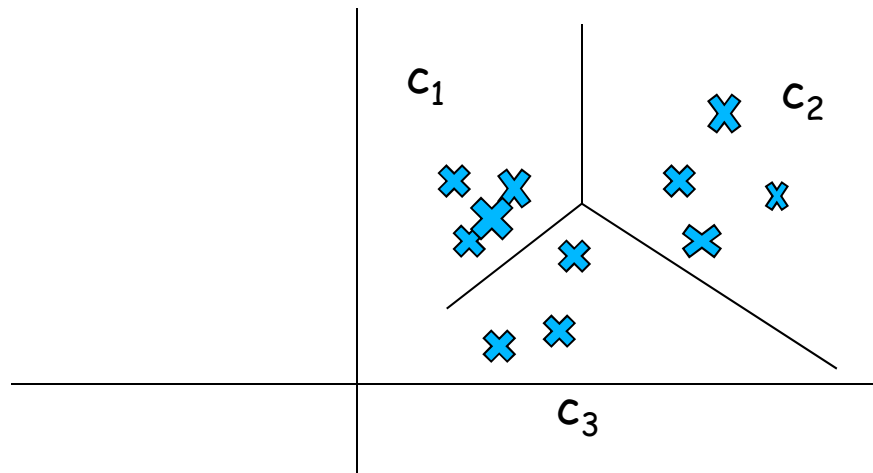


Fig. 3.16. The neurons, shown as circles, are labeled with the symbols of the phonemes with which they made the best match. Distinction of /k,p,t/ from this map is not reliable, and this phonemic class is indicated by symbol #. An analysis of the transient spectra of these phonemes by an auxiliary map is necessary.



Not very good for plosives

10ms interval - Quasiphonema

A sequence of quasiphonemes as phonoemes

Post- Processing: String matching

Isolated, word recognition with 1000 words.

Kohonen Learning:

- A high number of iterations needed(10,000 at least)
- N_c - neighborhood should be selected carefully.
- $\alpha(k)$ should start around 1 and should be decreased in time.

Supervised from unsupervised:

- Use labeled samples as unlabeled.
- Iterate the learning algorithm for clustering.
- Label samples and adjust borders accordingly.

www.ai-junkie.com/ann/som/som1.html

1. Augment and normalize input patterns to unit length.
$$x_i \leftarrow \{1, x_i\}, x_i \leftarrow x_i / \|x_i\|$$
2. Initialize all weights w_1, w_2, \dots, w_c to random values. (You may choose c random points from data and assign them as weights.)

3. When a new pattern X is presented, find

$$net_j = w_j^T X \quad \text{for all } 1 \leq j \leq c$$

4. Update the only weight with the highest net function.

5. Update rule: $w_j \leftarrow w_j + \eta X$

$$w_j \leftarrow w_j / \|w_j\|$$

6. Repeat 3-5 until no change occurs in w 's.

7. Return w_1, w_2, \dots, w_c .

8. The clusters are formed by assigning each pattern to w_j with highest net_j .