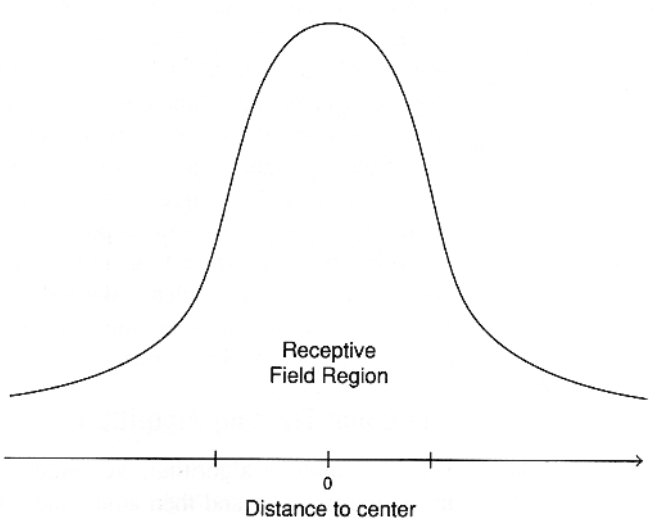
Radial Basis Functions

15-486/782: Artificial Neural Networks
David S. Touretzky

Fall 2006

Biological Inspiration for RBFs

The nervous system contains many examples of neurons with “local” or “tuned” receptive fields.



- Orientation-selective cells in visual cortex.
- Somatosensory cells responsive to specific body regions.
- Cells in the barn owl auditory system tuned to specific inter-aural time delays.

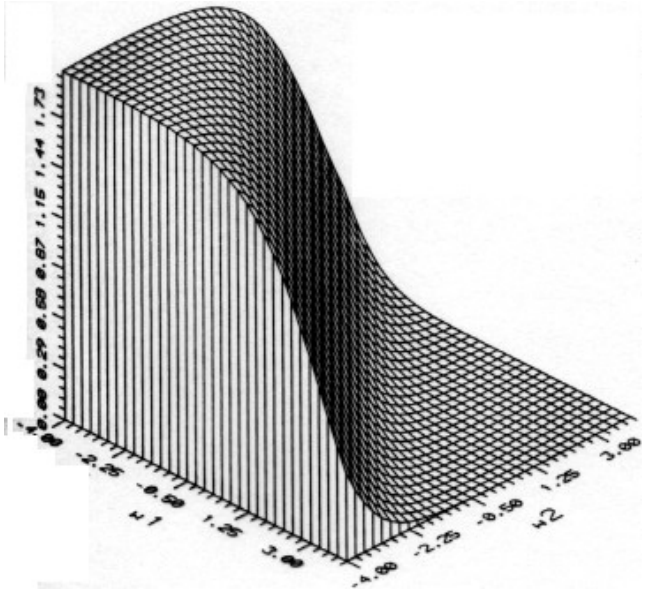
This local tuning is due to network properties.

Sigmoidal vs. Gaussian Units

Sigmoidal unit:

$$y_j = \tanh\left(\sum_i w_{ji} x_i\right)$$

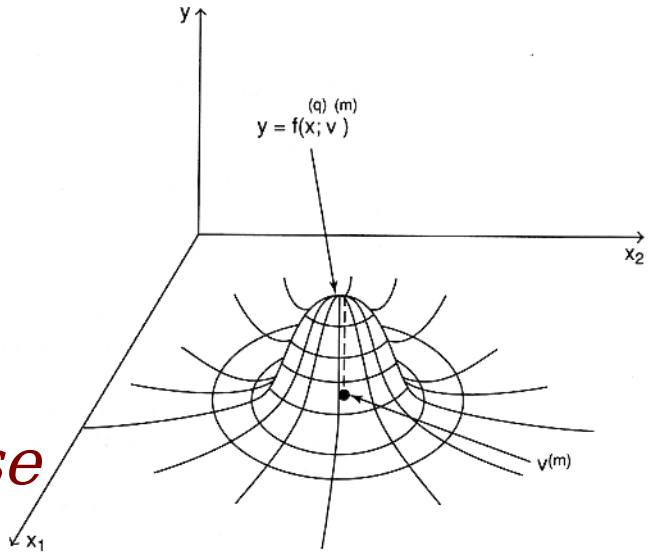
Decision boundary is a hyperplane



Gaussian unit:

$$y_j = \exp\left(\frac{-\|\vec{x}-\vec{\mu}_j\|^2}{\sigma_j^2}\right)$$

Decision boundary is a hyperellipse



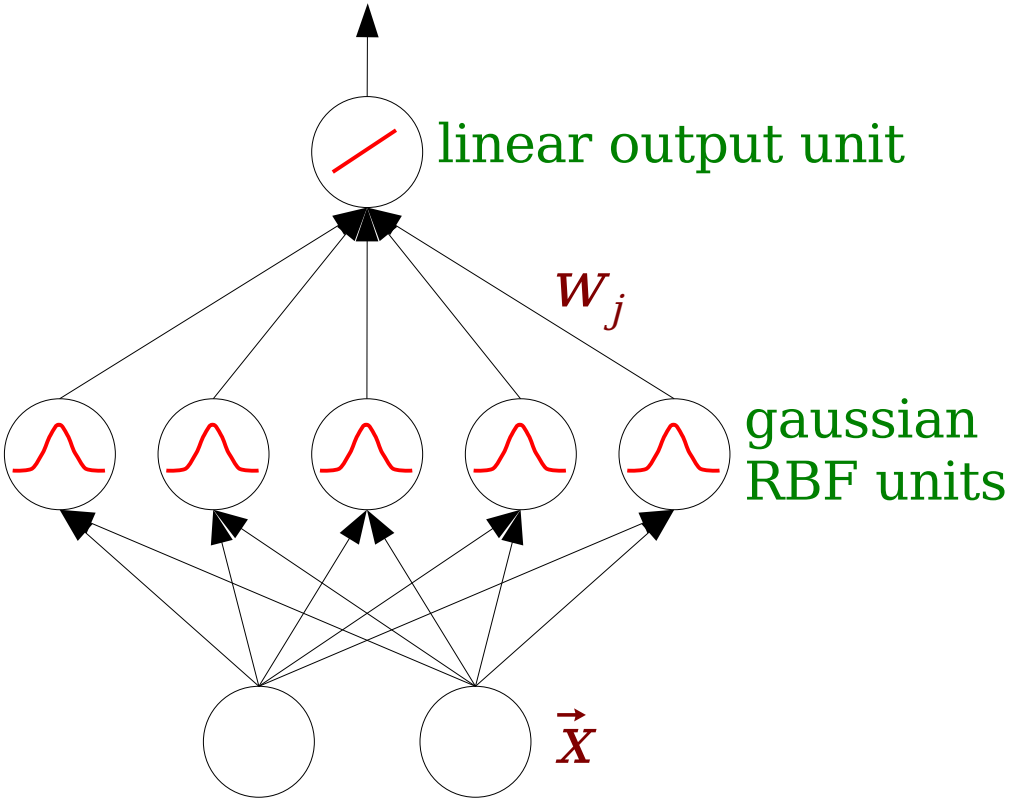
RBF = Local Response Function

Why do we use exp of distance squared: $\exp(-\|\boldsymbol{x}-\boldsymbol{\mu}\|^2)$
instead of dot product $\boldsymbol{x}\cdot\boldsymbol{w}$*?*

With dot product the response is linear along the preferred direction **w**, at all distances. Not local.

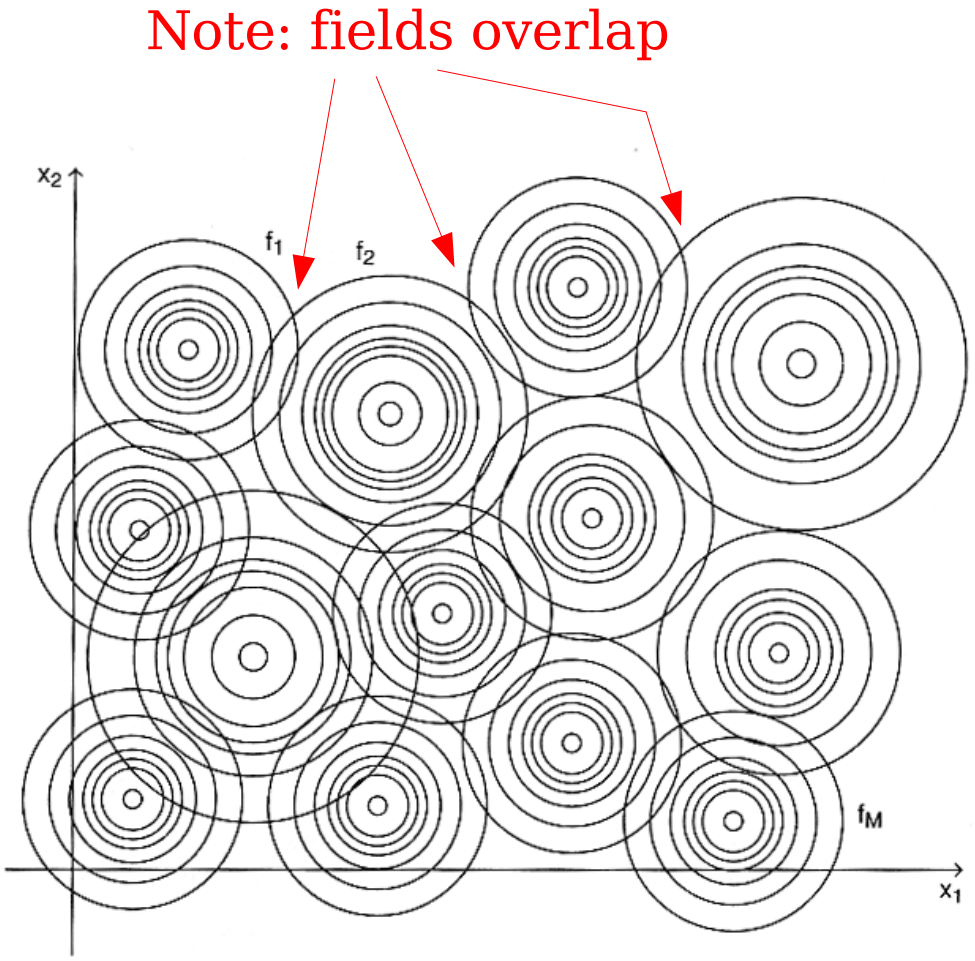
If we want <u>local</u> units, we must use distance instead of dot product to compute the degree of "match".

RBF Network



$$Output = \sum_j w_j \cdot \exp\left(\frac{-\|\vec{x}-\vec{\mu}_j\|^2}{\sigma_j^2}\right)$$

Tiling the Input Space



Properties of RBF Networks

Receptive fields overlap a bit, so there is usually more than one unit active.

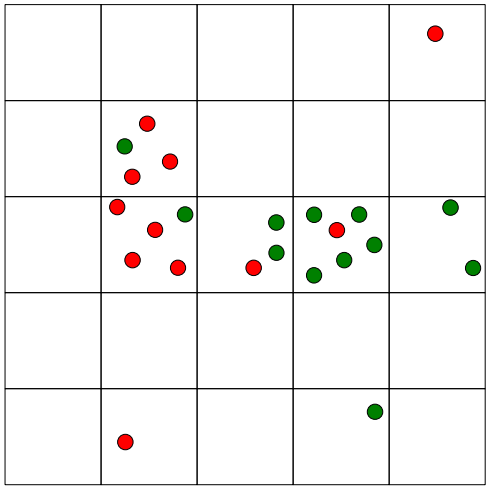
But for a given input, the total number of active units will be small.

The locality property of RBFs makes them similar to Parzen windows.

Multiple active hidden units distinguishes RBF networks from competitive learning or counterpropagation networks, which use winner-take-all dynamics.

RBFs and Parzen Windows

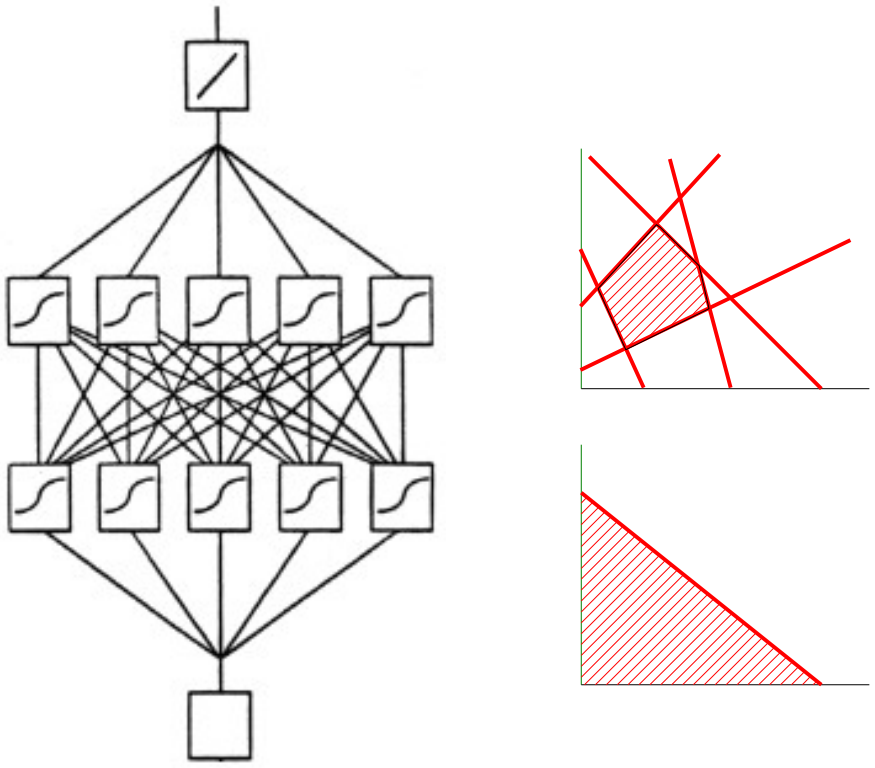
The locality property of RBFs makes them similar to Parzen windows.



Calculate the local density of each class and use that to classify new points within the window.

Build Our Own Bumps?

Two layers of sigmoidal units can be used to synthesize a “bump”. But it's simpler to use gaussian RBF units.



Training an RBF Network

1. *Use unsupervised learning to determine a set of bump locations $\{\vec{\mu}_j\}$, and perhaps also $\{\sigma_j\}$.*

2. *Use LMS algorithm to train output weights $\{w_j\}$.*

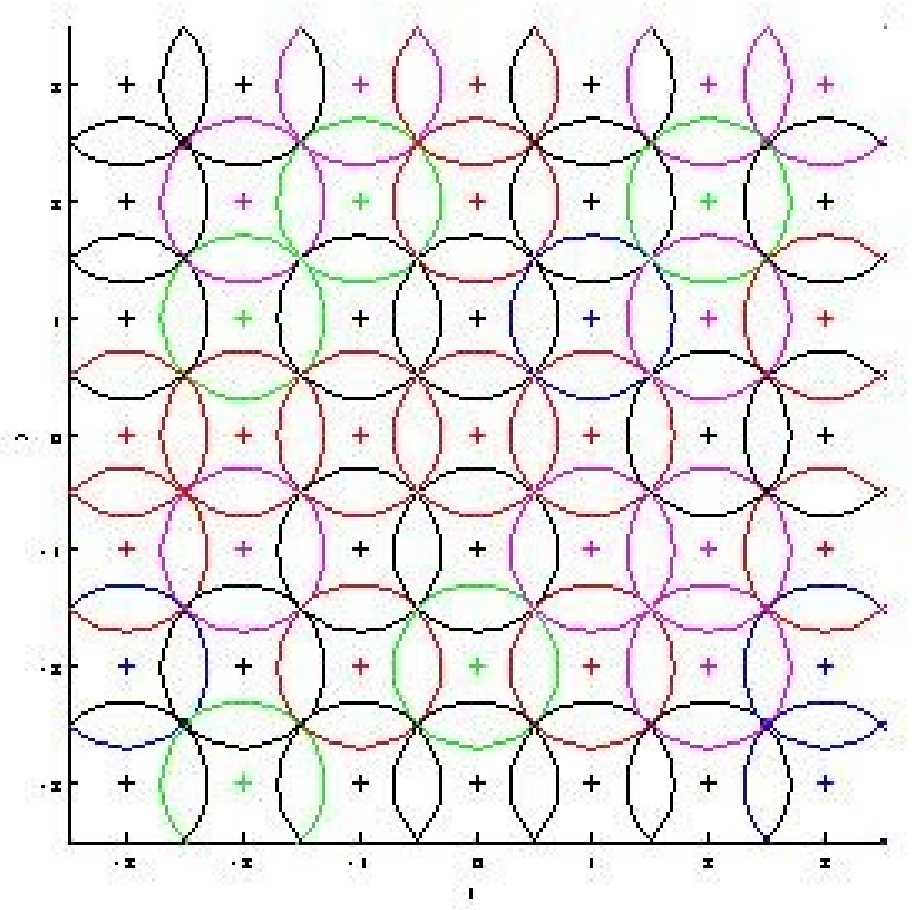
This is a hybrid training scheme.

Training is very fast, because we don't have to back-propagate an error signal through multiple layers.

Error surface is quadratic: no local minima for the LMS portion of the algorithm

RBF Demo

matlab/rbf/rbfdemo



Regularly spaced gaussians with fixed σ^2

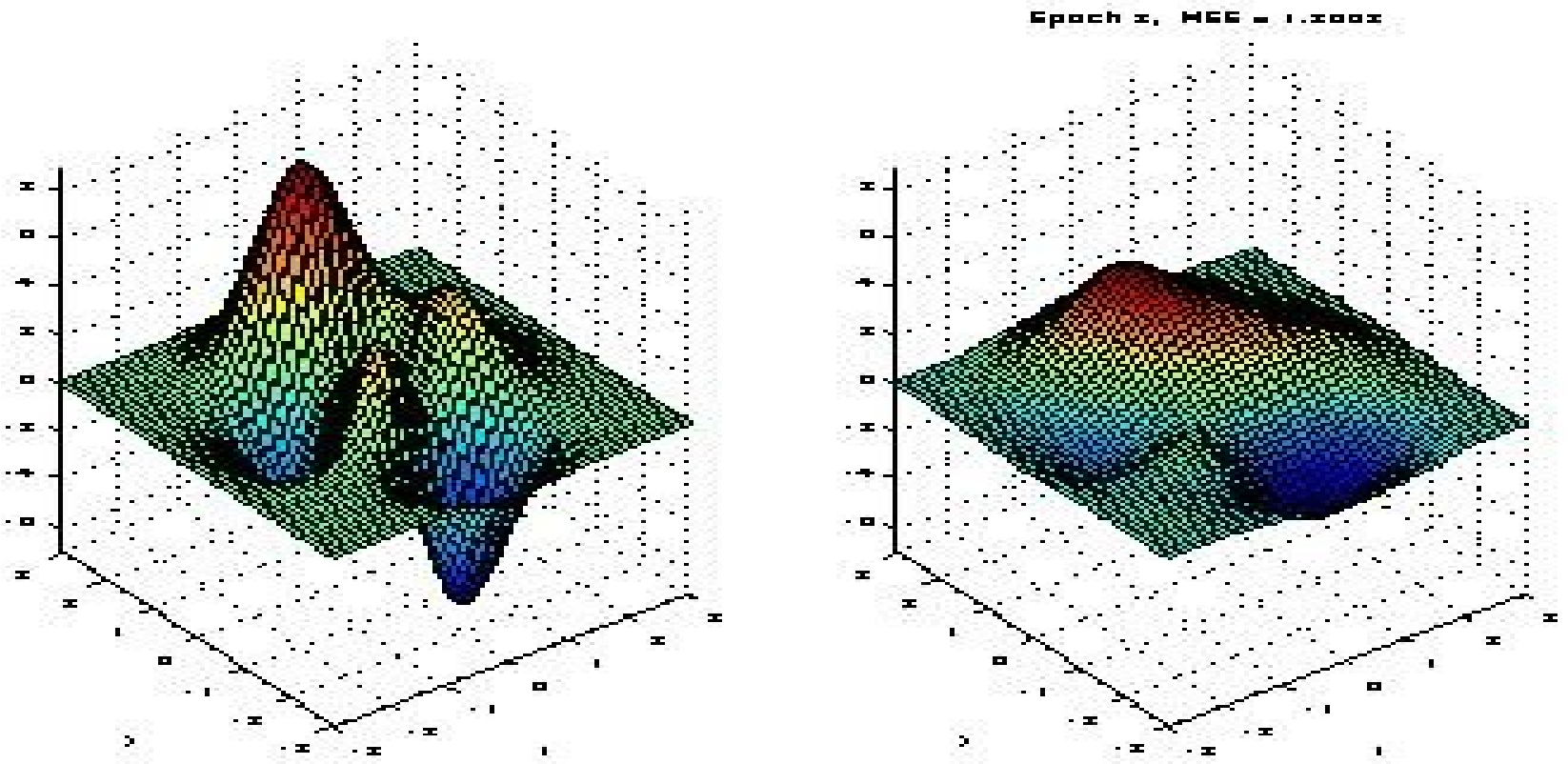
Training Tip

Since the RBF centers and variances are fixed, we only have to evaluate the activations of the RBF units once.

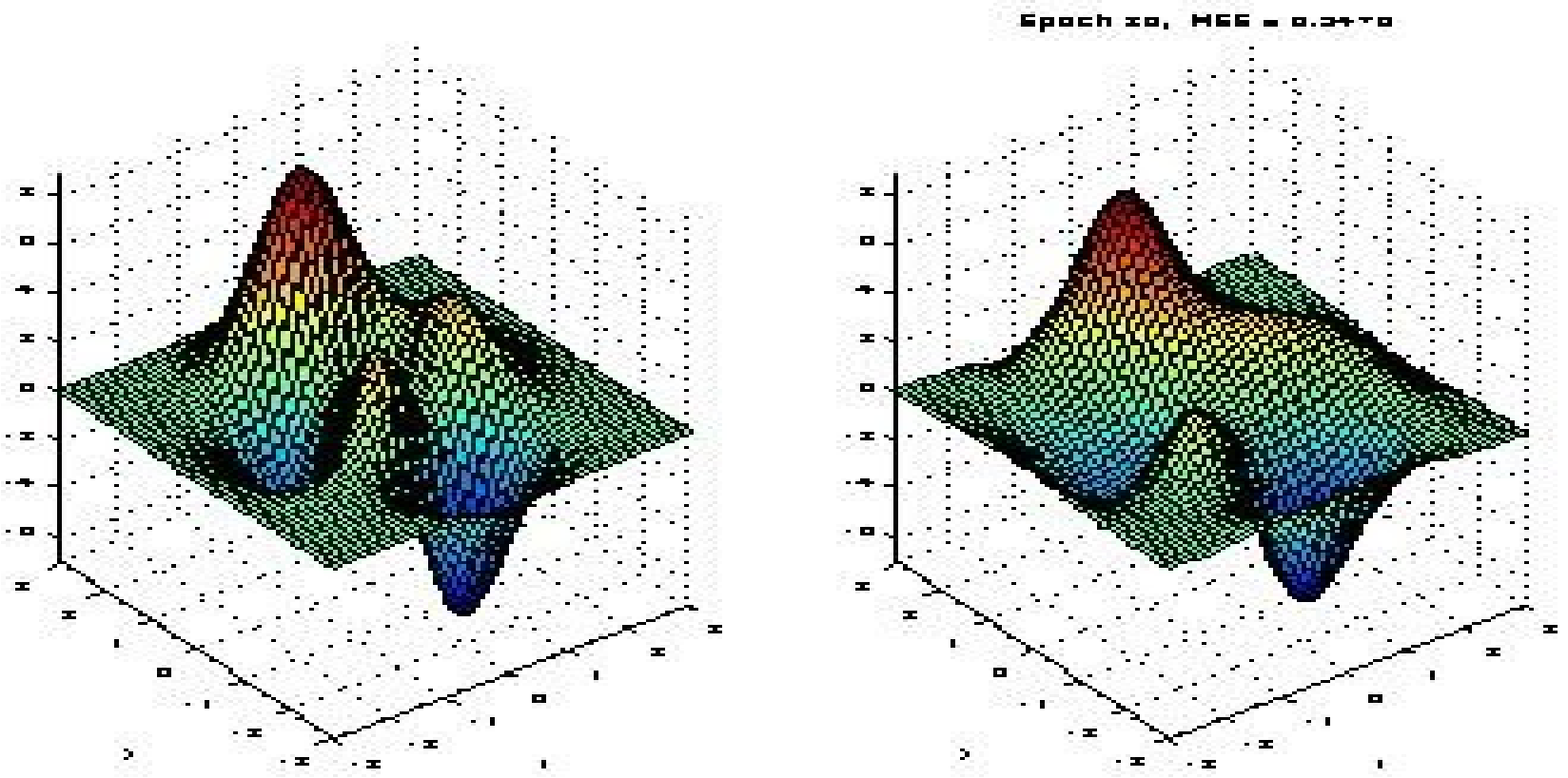
Then train the RBF-to-output weights interatively, using LMS.

Learning is very fast.

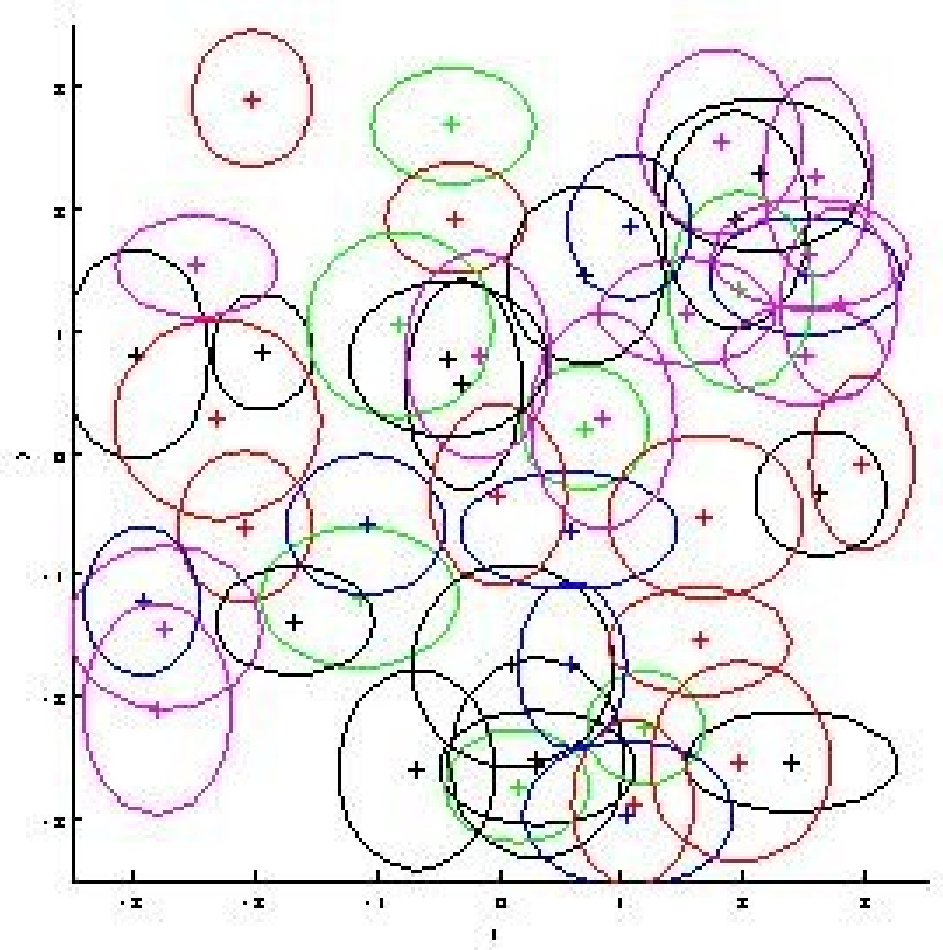
Early in Training



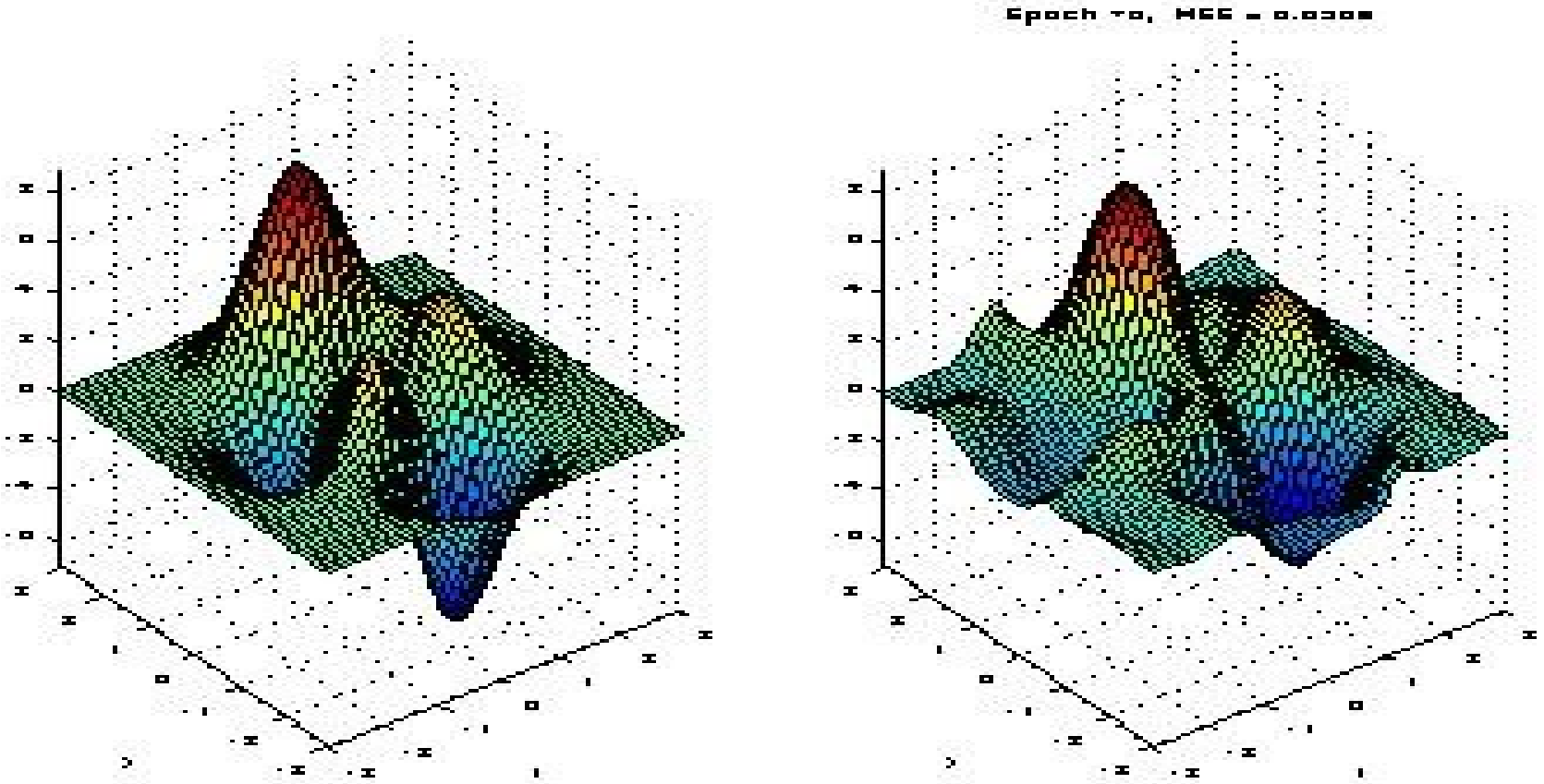
Training Complete



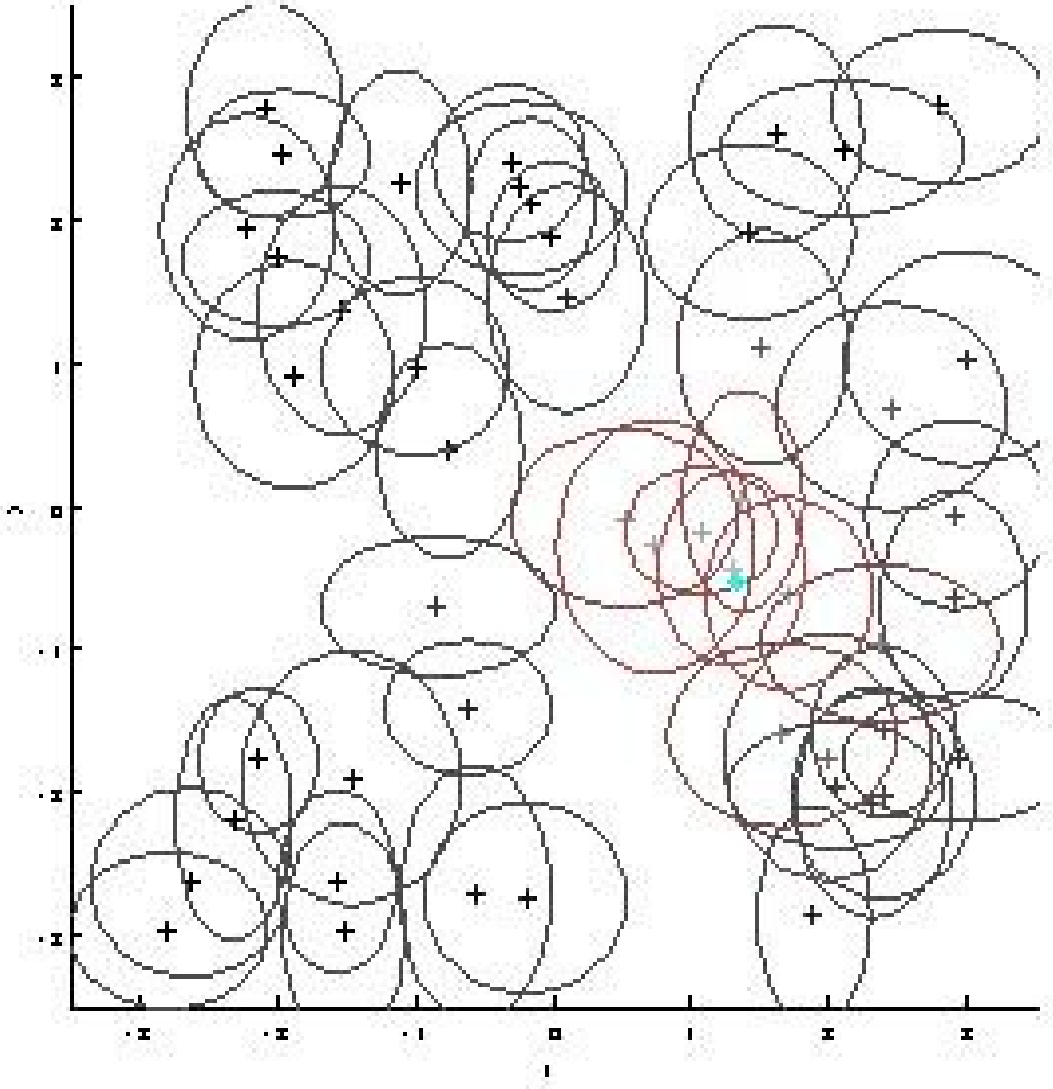
Random Gaussians



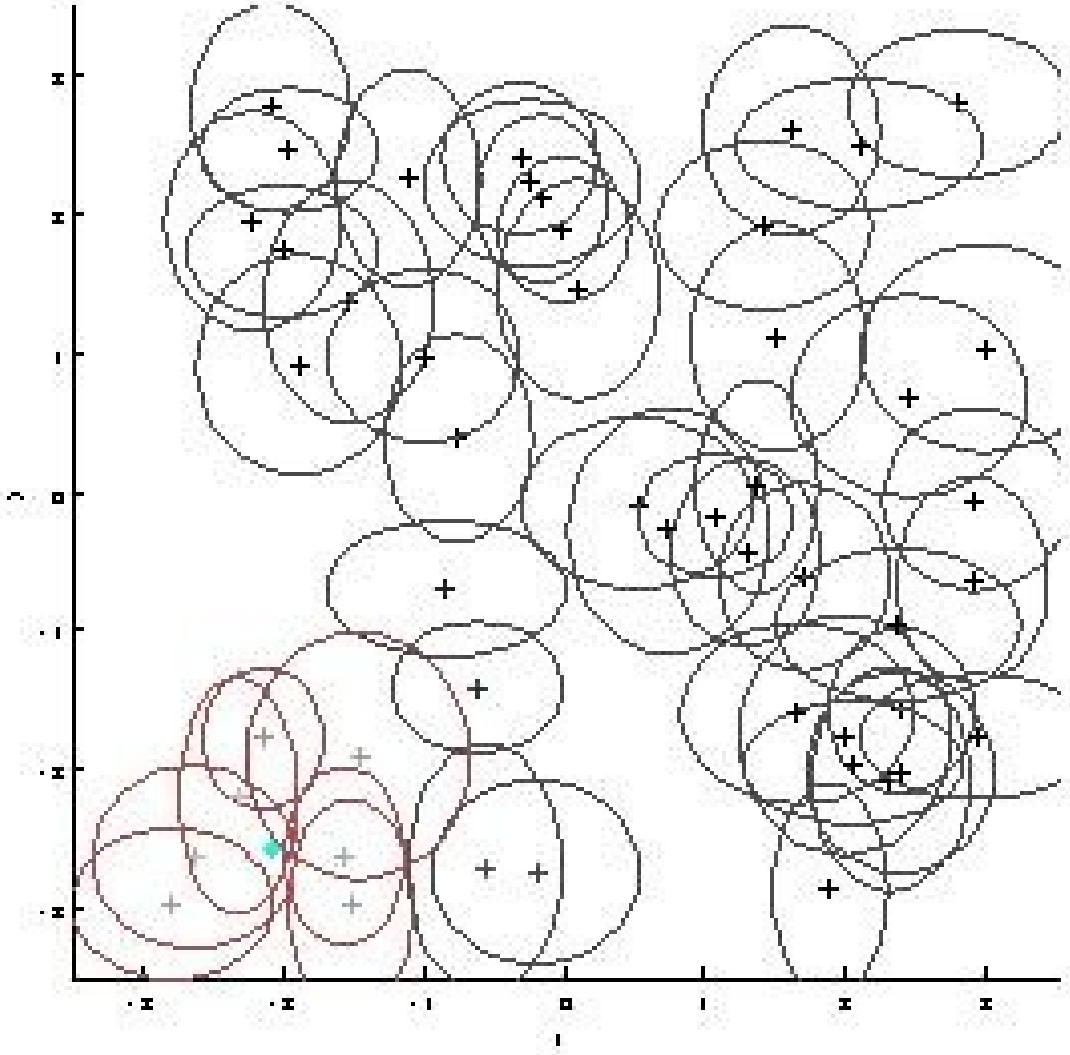
After Training



Locality of Activation



Locality of Activation



Winning in High Dimensions

RBFs really shine for low-dimensional manifolds embedded in high dimensional spaces.

In low dimensional spaces, can just use a Parzen window (for classification) or a table-lookup interpolation scheme.

But in high dimensional spaces, we can't afford to tile the entire space. (Curse of dimensionality.)

We can place RBF units only where they're needed.

How to Place RBF Units?

1) Use k-means clustering, intialized from randomly chosen points from the training set.

2) Use a Kohonen SOFM (Self-Organizing Feature Map) to map the space. Then take selected units' weight vectors as our RBF centers.

k-Means Clustering Algorithm

1) Choose k cluster centers in the input space. (Can choose at random, or choose from among the training points.)

2) Mark each training point as “captured” by the cluster to which it is closest.

3) Move each cluster center to the mean of the points it captured.

4) Repeat until convergence. (Very fast.)

Online Version of k-Means

1. Select a data point $\bar{x}_i$.

2. Find nearest cluster; its center is at $\bar{\mu}_j$.

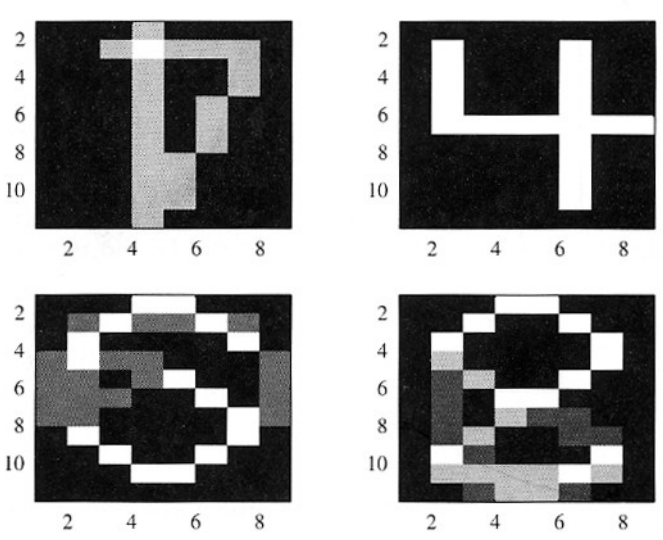
3. Update the center:

$$\Delta \bar{\mu}_j \leftarrow \eta\left(\bar{x}_i - \bar{\mu}_j\right)$$

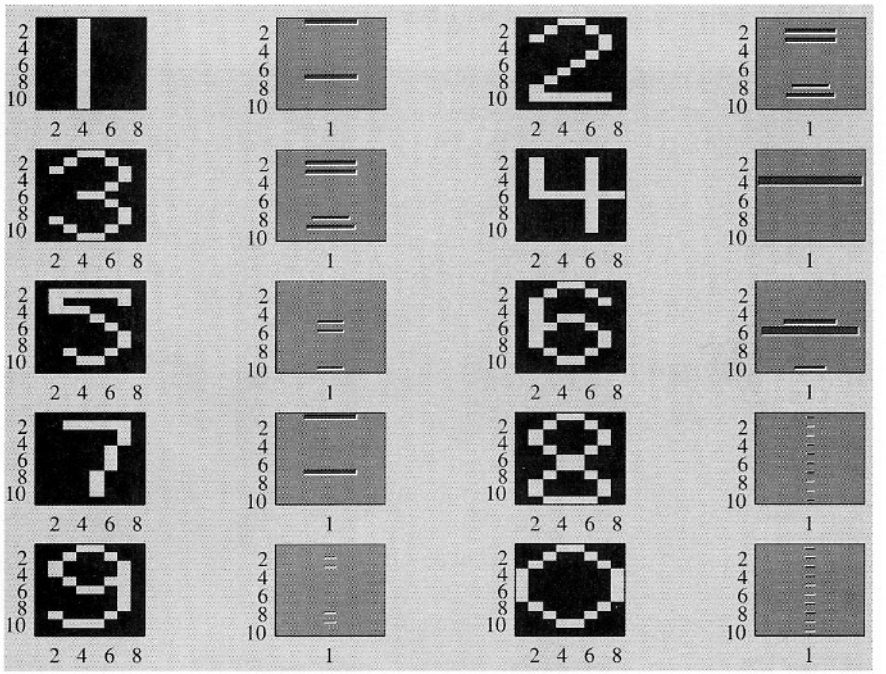
where eta = 0.03 *(learning rate)*

This is on-line competitive learning.

Recognizing Digits (16x16 pixels)

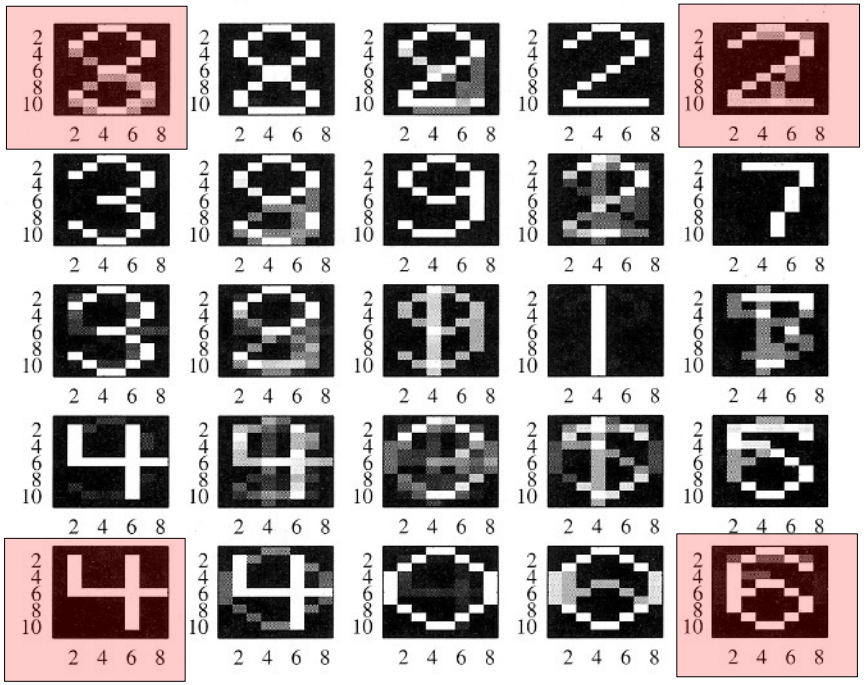


Four RBF centers trained by k-means clustering.

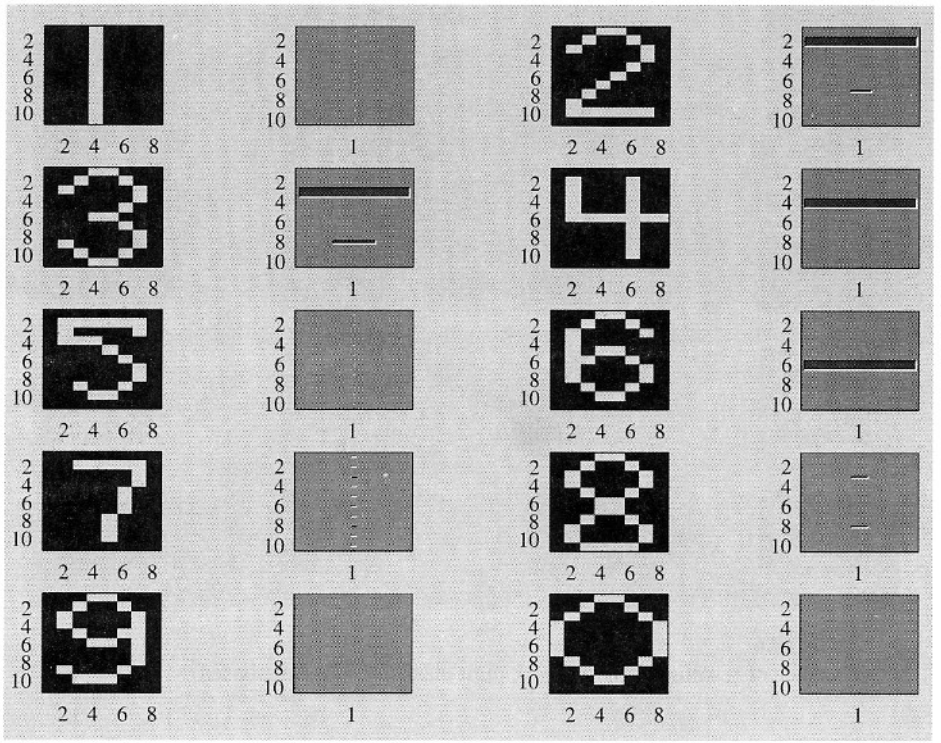


Only the 4 and the 6 are recognized. Classifier performance is poor. Not a good basis set.

Using SOFM to Pick RBF Centers



Train a 5x5 Kohonen feature map. Then take the four corner units as our RBF centers.



Performance is better. Recognizes 2, 3, 4, 6.

Determining the Variance σ^2

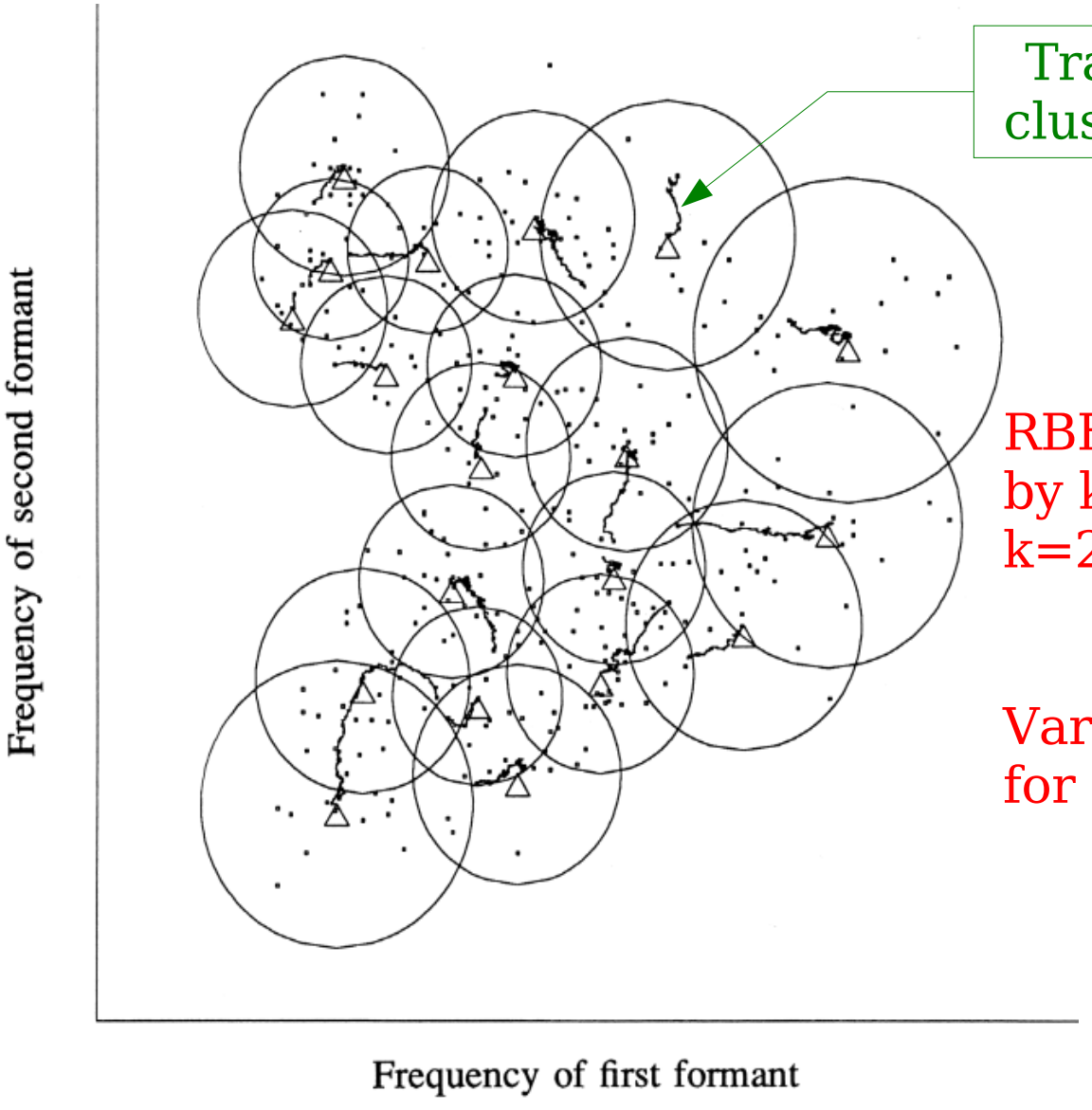
1) Global “first nearest neighbor” rule:

σ = mean distance between each unit j and its closest neighbor.

1) P-nearest-neighbor heuristic:

Set each σ_j so that there is a certain amount of overlap with the P closest neighbors of unit j.

Phoneme Clustering (338 points)



RBF centers set by k-means: k=20.

Variances set for overlap P=2.

Phoneme Classification Task

- Moody & Darken (1989): classify 10 distinct vowel sounds based on F1 vs. F2.
- 338 training points; 333 test points.

# Gaussian Units	20	40	60	80	100
% Error on Test Set	26.7%	24.9%	21.3%	19.5%	18.0%

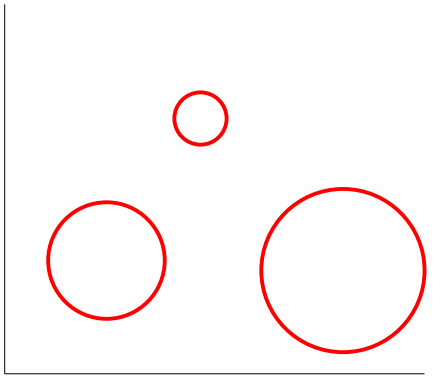
- Results comparable to those of Huang & Lippmann:

Classification Method	% Error on Test Set	Number of Training Tokens
K Nearest Neighbors (Non-Adaptive)	18.0%	338
Gaussian Classifier (Non-Adaptive)	20.4%	338
2-Layer Back Prop (Adaptive)	19.8%	50,000
Feature Map, 100 Nodes (Adaptive)	22.8%	10,000 (Feature Map Nodes) 50 (Output Nodes)

Defining the Variance

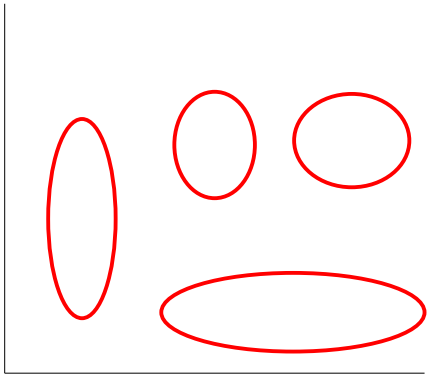
Radially symmetric fields:

$$d_j = \frac{\|\vec{x} - \vec{u}_j\|^2}{\sigma_j^2}$$



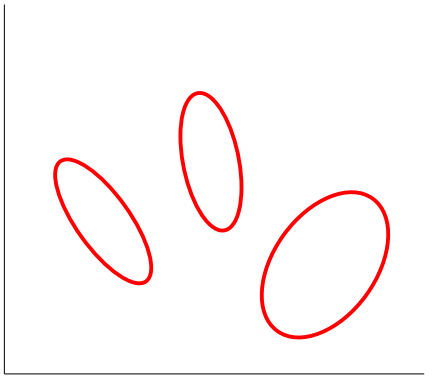
Elliptical fields, aligned with axes:

$$d_j = \sum_i \frac{\left(x_i - \mu_{ji}\right)^2}{\sigma_{ji}^2}$$



Arbitrary Elliptical Fields

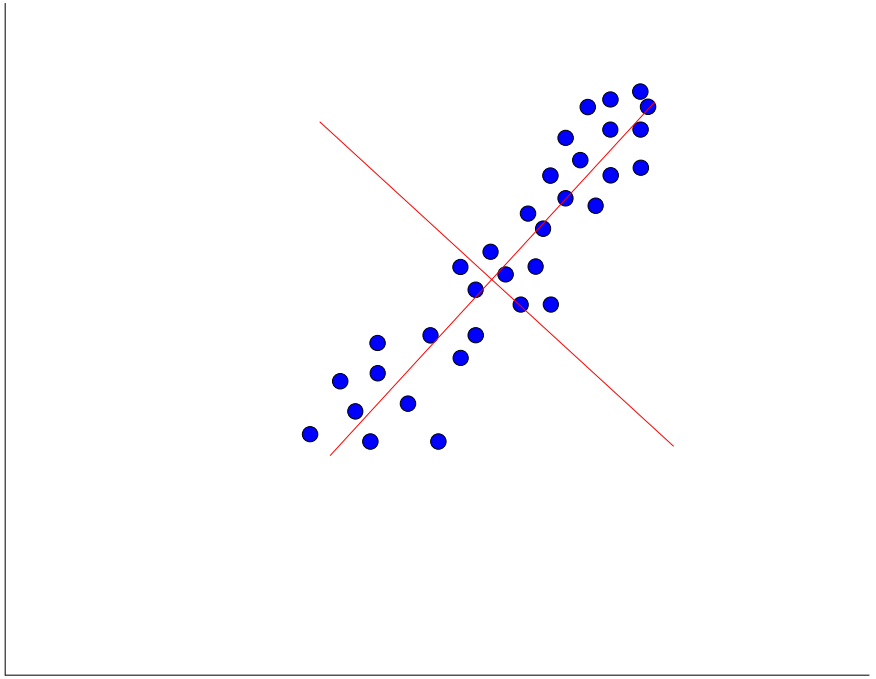
Requires co-variance matrix Σ
with non-zero off-diagonal terms.



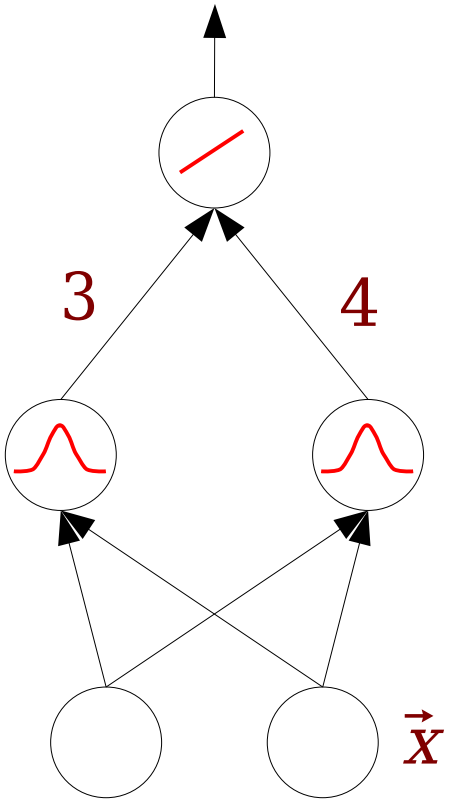
For many pattern recognition tasks, we can re-align the axes with PCA and normalize the variances in a pre-processing step, so a simple set of $\{\sigma_j\}$ values suffices.

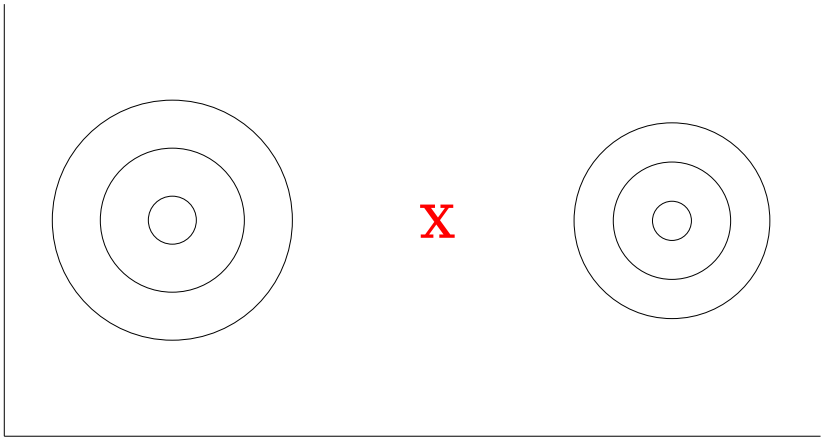
Transforming the Input Space

Principal Components Analysis transforms the coordinate system. Now ellipses can be aligned with the major axes.



Smoothness Problem



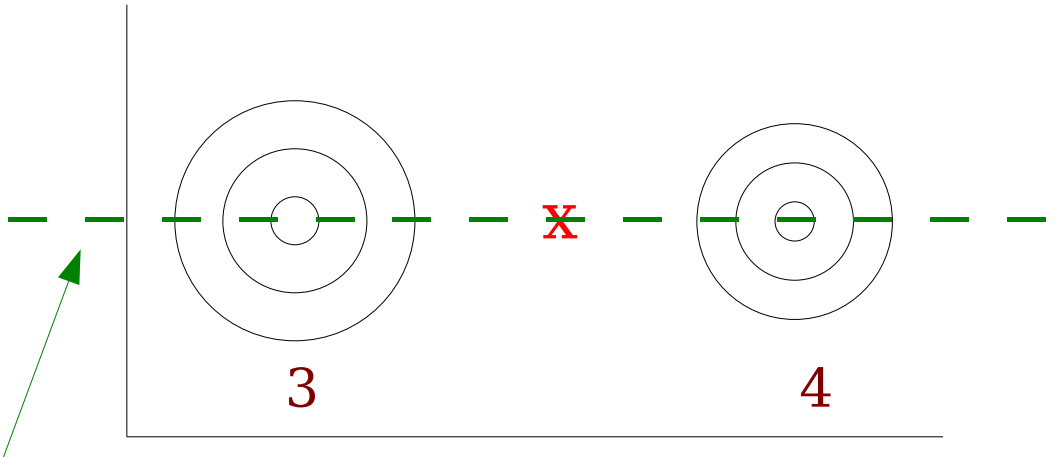


At point x neither RBF unit is very active, so the output of the network sags close to zero. Should be 3.5.

Assuring Smoothness

To assure smoothness, we can normalize the output by the total activation of the RBF units.

$$Output = \frac{\sum_j y_j \cdot w_j}{\sum_j y_j}$$



Smooth interpolation along this line.
No output sag in the middle.

Training RBF Nets with Backprop

Calculate $\frac{\partial E}{\partial \mu_j}$, $\frac{\partial E}{\partial \sigma_j}$, *and* $\frac{\partial E}{\partial w_j}$.

Update all parameters in parallel.

Problems:

– Slow!

– σ's can grow large: unit no longer “locally” tuned.

Advantage:

– Units are optimally tuned for producing correct outputs from the network.

Summary of RBFs

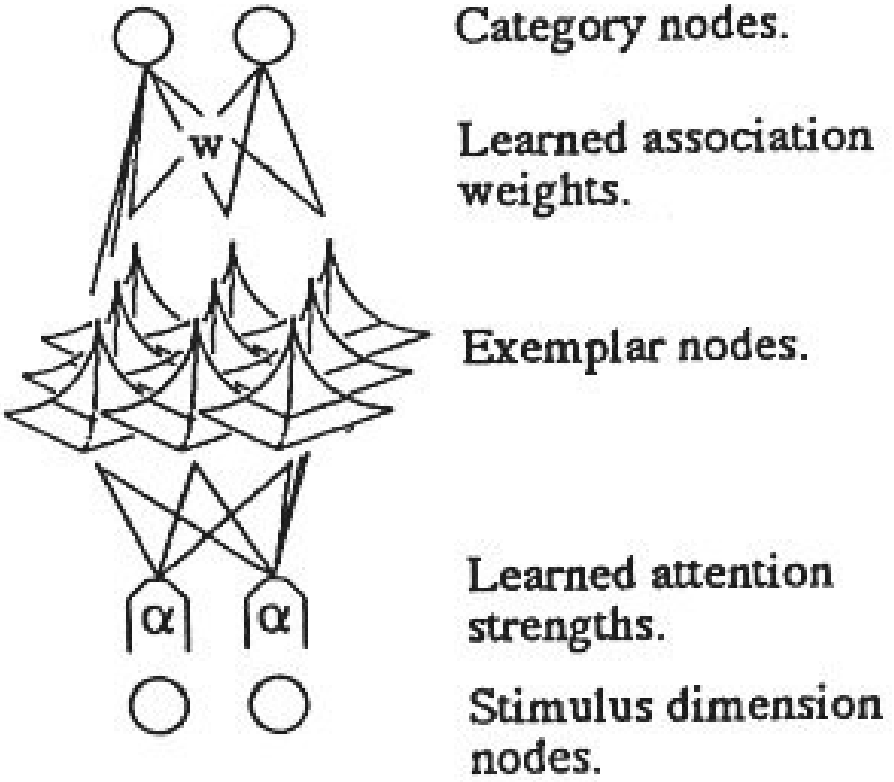
- RBF units provide a new basis set for synthesizing an output function. The basis functions are not orthogonal and are overcomplete.
- RBFs only work well for smooth functions.
  - Would not work well for parity.
- Overlapped receptive fields give smooth blending of output values.
- Training is much faster than backprop nets: each weight layer is trained separately.

Summary of RBFs

- Hybrid learning algorithm: unsupervised learning sets the RBF centers; supervised learning trains the hidden to output weights.
- RBFs are most useful in high-dimensional spaces. For a 2D space we could just use table lookup and interpolation.
- In a high-D space, curse of dimensionality important.
  - OCR: 16 x 16 pixel image = 256 dimensions.
  - Speech: 5 frames @ 16 values/frame = 80 dimensions.

Psychological Model: ALCOVE

John Kruschke's ALCOVE (Attention Learning Covering Map) models category learning with an RBF network.



Category Learning

• Train humans on a toy classification problem. Then measure their generalization behavior on novel exemplars.

• ALCOVE: each training example defines a Gaussian.

• All variances equal.

• Output layer trained by LMS.

ALCOVE Equations

$$\textit{Hiddens: } a_j^{hid} = \exp\left[-c\cdot\left(\sum_j \alpha_i |h_{ji} - \alpha_i^{in}|^r\right)^{q/r}\right]$$

c is a specificity constant; α_i is attentional strength

$$\textit{Category: } a_k^{out} = \sum_j w_{kj} a_j^{hid}$$

$$\textit{Response: } Pr(K) = \exp(\phi\, a_K^{out}) / \sum_j \exp\left(\phi\, a_j^{out}\right)$$

ϕ is a mapping constant

(softmax)

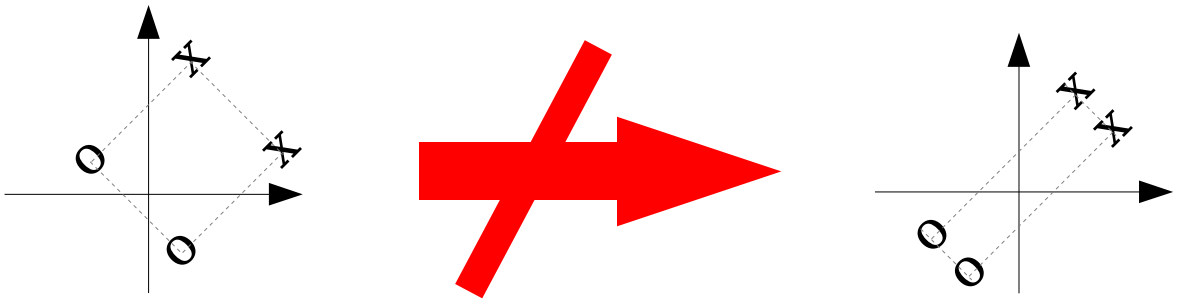
Dimensional Attention

Emphasize dimensions that distinguish categories, and de-emphasize dimensions that vary within a category.

Makes the members of a category appear more similar to each other, and more different from non-members.

Adjust dimensional attention α_i *based on* $\partial E / \partial \alpha_i$

Dimensional Attention



Because ALCOVE does not use a full covariance matrix, it cannot shrink or expand the input space along directions not aligned with the axes. However, for cognitive modeling purposes, a diagonal covariance matrix appears to suffice.

Disease Classification Problem

	Exemplar	Symptoms		Observed	ALCOVE	Config cue
Terrigitis	T1	1 1 1 1		.88	.82	.76
	T2	0 1 1 1		.89	.78	.76
	T3	1 1 0 0		.73	.82	.76
	T4	1 0 0 0		.77	.78	.76
Midosis	M1	1 0 1 0		.12	.22	.25
	M2	0 0 1 0		.17	.18	.25
	M3	0 1 0 1		.25	.22	.25
	M4	0 0 0 1		.33	.18	.25
Novel Items: Test Set	N1	0 0 0 0	T	.53	.59	.44
	N2	0 0 1 1	T	.53	.59	.44
	N3	0 1 0 0	T	.75	.64	.46
	N4	1 0 1 1	T	.67	.64	.46
	N5	1 1 1 0	M	.45	.41	.58
	N6	1 1 0 1	M	.38	.41	.58
	N7	0 1 1 0	M	.36	.36	.55
	N8	1 0 0 1	M	.28	.36	.55

Humans and ALCOVE: N3,N4 > N1,N2 and N5,N6 > N7,N8