High Order Eigentensors as Symbolic Rules in Competitive Learning

Hod Lipson[1] and Hava T. Siegelmann[2]

[1] Mechanical Engineering Department
[2] Industrial Engineering and Management Department
Technion – Israel Institute of Technology, Haifa 32000, Israel
Current e-mail: hlipson@mit.edu, iehava@ie.technion.ac.il

Abstract. We discuss properties of high order neurons in competitive learning. In such neurons, geometric shapes replace the role of classic 'point' neurons in neural networks. Complex analytical shapes are modeled by replacing the classic synaptic weight of the neuron by high-order tensors in homogeneous coordinates. Such neurons permit not only mapping of the data domain but also decomposition of some of its topological properties, which may reveal symbolic structure of the data. Moreover, eigentensors of the synaptic tensors reveal the coefficients of polynomial rules that the network is essentially carrying out. We show how such neurons can be formulated to follow the maximum-correlation activation principle and permit simple local Hebbian learning. We demonstrate decomposition of spatial arrangements of data clusters including very close and partially overlapping clusters, which are difficult to separate using classic neurons.

1 Introduction

A phase diagram contains data points representing measurements of a phenomenon plotted in multi-dimensional space. If the measured phenomenon follows no rules, the data points will uniformly fill the data space. However, if there are any governing principles to the measured phenomenon, the data points will not fill the space uniformly, but rather will form some kind of structure or pattern. Tools for exploratory data analysis such as neural networks may then be used to map this structure, and so create a model for the behavior of the phenomenon.

In many cases, however, mapping the phenomenon, i.e. determining what areas in the data space it is more likely to occupy, is not sufficient. In order to understand the *laws* that govern the phenomenon, it is necessary to decompose the mapped volume into its components and derive relationships among them. It is convenient to associate the simpler *mapping* of the phenomenon with determination of its *metrics*, and the "deeper" understanding of the *governing principles* or *symbolic structure* with determination of its *topology*. In this paper we take the view where in order to extract symbolic meaning from an observed phenomenon it is necessary to use neuronal units that can individually account for a complete symbolic rule. In practice, we use geometric shapes: the shape itself is the symbolic rule and its parameters (say curvatures and gradients) are the tunable parameters. We describe an augmentation to classic competitive neurons that permits them to decipher these aspects.

The ability to determine the topological structure of the data domain is considered a primary capacity of Kohonen's self-organizing map (SOM) (Kohonen, 1997). Under certain conditions, a SOM network may self-organize so that each neuron relocates to the center of a cluster of input points which it represents. Due to the connectivity of the net, the topology of the net is also mapped onto the topology of the data domain, thus revealing topological properties of the structure such as cluster proximity. However, the topological structure of the net may also act as a *constraint* on the arrangement of the neurons, impeding its ability to capture certain configurations. When this topological constraint is released by adopting full connectivity, a general form of a vector quantization (VQ) clustering algorithm is obtained. Such algorithms (Pal *et al*, 1993) can map any data arrangement, but they do not provide information regarding the topological structure. Similarly, most other networks acquire topological information implicitly into their weights; this information cannot be directly extracted.

In this paper we explore an alternative approach to modeling the topology of the data domain. Modeling is achieved by using neurons that not only map the location of their corresponding data, but also explicitly map its local topological and geometrical properties. These 'geometric neurons' are of higher dimensionality than their input domain, and may therefore track *features* of the activation area that might correspond to local symbolic properties. They use high-order 'synaptic tensors' instead of classic synaptic weight vectors, where weights correspond also to *combinations* of inputs of various orders. When these neurons are used in a network configuration, local topological properties are accumulated to *explicitly* reveal the global topological arrangement of the data. These neurons obey the simple Hebbian-type learning rule, and, depending on their shape and base functions, can reveal configurations even among close and partially overlapping clusters.

The paper first outlines the concept of the proposed enhancement in light of existing work, and then provides a mathematical formulation for analytic geometric neurons with polynomial base functions. We show that a classic neuron is a first-order case of the geometric neuron, and the second-order neuron corresponds to the well-established ellipsoidal (Mahalanobis) metric neuron. We describe the neuron itself, its activation, its learning scheme and then demonstrate its functionality within a net.

2 Shape-Sensitive Neurons

The fundamental property of a shape sensitive neuron is that it is capable of mapping the *local* topological and geometrical properties of a data volume because, unlike a point neuron, it has topological and geometric properties of its own. These properties are parametric and hence adaptive. Unlike classic networks, the topological structure of the data is then directly accessible, since the topology of each neuron is known and simple. A schematic illustration of a network with geometric neurons is shown in Figure 1(a), where point neurons are replaced by higher-order 'blob' neurons which can take the form of various topological components such as links, forks and volumes, as well as of ordinary point neurons. Four actual shape sensitive neurons are shown in Figure 1(b).

High order neurons are defined as neurons which accept input not only from single inputs, but also from *combinations* of inputs, such as sets of inputs multiplied to various orders. The use of high order neurons in general is not new. High order neurons are generally associated with more degrees of freedom rather than explicit topological properties. Explicit geometric properties have been introduced for specific cases of prototype-based clustering and competitive neural networks. Gustafson and Kessel (1979) used the covariance matrix to capture ellipsoidal properties of clusters. Davé (1989) used fuzzy clustering with a non-Euclidean metric to detect lines in images. This concept was later expanded, and Krishnapuram *et al* (1995) used general second-order shells such as ellipsoidal shells and surfaces. For an overview and comparison of these methods see Frigui and Krishnapuram (1996). Incorporation of Mahalanobis (elliptical) metrics in neural networks was addressed by Kavuri and Venkatasubramanian (1993) for fault analysis applications, and by Mao and Jain (1996) as a general competitive network with embedded principal component analysis units. Kohonen uses adaptive tensorial weights (Kohonen, 1997) to capture significant variances in the components of input signals, thereby introducing a weighted Euclidean distance in the matching law. Abe *et al* (1997) attempt to extract fuzzy rules using ellipsoidal units and compare successfully to other rule-extracting methods. In this paper we explore the possibility of using neurons with general and explicit geometric properties under direct Hebbian learning. These neurons are not limited to ellipsoidal shapes.

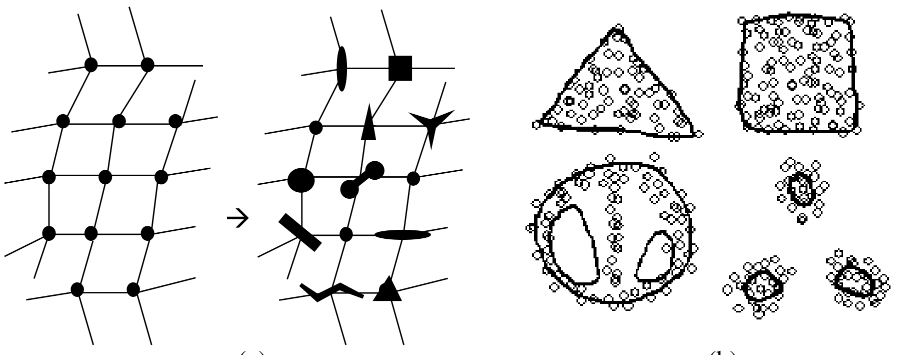

Fig. 1. (a) A schematic illustration of a network with geometric neurons. Point neurons are replaced by higher-order 'blob' neurons that can take the form of various topological (symbolic) components such as links, forks and volumes, as well as of ordinary point neurons. (b) Examples of four high order geometric neurons developed in this work, with different geometric and topological structures, and the data points they represent.

In the following sections we adopt a notation where: $\boldsymbol{x}, \boldsymbol{w}$ are column vectors, $\boldsymbol{W}, \boldsymbol{D}$ are matrices; $\boldsymbol{x}_H$, $\boldsymbol{W}_H$ denote vectors and matrices in homogeneous representation (described later), $\boldsymbol{x}^{(j)}$, $\boldsymbol{D}^{(j)}$ - the jth vector/matrix corresponding to the jth neuron/class; $\boldsymbol{x}_1$, $\boldsymbol{D}_{ij}$ - element of a vector/matrix; m - the order of neuron; d - the dimensionality of the input; N - the size of the layer (the number of neurons).

3 A High-Order Neuron with Polynomial Base Functions

In classical self-organizing networks, each neuron *j* is assigned a synaptic weight vector $\boldsymbol{w}^{(j)}$. The winning neuron *j(x)* in response to an input $\boldsymbol{x}$ is the one showing the highest *correlation* with the input, i.e. neuron *j* for which $\boldsymbol{w}^{(j)T}\boldsymbol{x}$ is the largest. Note that when the synaptic weights $\boldsymbol{w}^{(j)}$ are normalized to a constant Euclidean length, then the above criterion becomes identical to the minimum Euclidean distance matching criterion. However, the use of a minimum-distance matching criterion incorporates several difficulties. The minimum distance criterion implies that the features of the input domain are spherical, i.e., matching deviations are considered equally in all directions, and distances between features must be larger than the distances between points within a feature.

These aspects preclude the ability to detect higher order, complex or ill-posed feature configurations and topologies, as these are based on higher geometrical properties such as *directionality* and *curvature*. This constitutes a major difficulty, especially when the input is of high dimensionality, where such configurations are difficult to visualize. Complex clusters may require complex metrics for separation.

The modeling constraints imposed by the maximum correlation matching criterion stem from the fact that the neuron's synaptic weight $\boldsymbol{w}^{(j)}$ has the same dimensionality as the input $\boldsymbol{x}$, i.e., the same dimensionality as a *single* point in the input domain, while in fact the neuron is modeling a *cluster* of points which may have higher order attributes such as directionality and curvature. We shall therefore refer to the classic neuron as a 'first-order' (zero degree) neuron, due to its correspondence to a *point* in multidimensional space.

Second Order

To circumvent this restriction, we augment the neuron with the capacity to map additional geometric and topological information. For example, as the first-order is a point neuron, the second-order case will correspond to orientation and size components, effectively attaching a local oriented coordinate system with non-uniform scaling to the neuron center and using it to define a new distance metric. Thus each second-order neuron will represent not only the mean value of the data points in the cluster it is associated with, but also the principal directions of the cluster and the variance of the data points along these directions. Intuitively, we can say that rather than defining a sphere, the second-order distance metric now defines a multi-dimensional oriented ellipsoid. After some mathematical manipulation (Gustafson and Kessel, 79), the second order information (orientation and scaling) can be shown to reside entirely in the correlation matrix $\boldsymbol{R}$ of the zero-mean data, and the matching criterion becomes

$$i(\mathbf{x}) = \arg_j \min\left[\left(\mathbf{x}-\mathbf{w}^{(j)}\right)^T \mathbf{R}^{-1}\left(\mathbf{x}-\mathbf{w}^{(j)}\right)\right], \quad j = 1,2,\ldots,N \qquad \textbf{(1)}$$

This criterion also corresponds to the term used in the maximum likelihood Gaussian classifier (Duda and Hart, 73). As it stands, Eq. (1) requires separate tracking the orientation and size information, in $\boldsymbol{R}^{(j)}$, and the position in $\boldsymbol{w}^{(j)}$. For a more systematic treatment, we combine these two coefficients into one *expanded covariance* denoted $\boldsymbol{R}_{\mathrm{H}}^{(j)}$ in *homogenous coordinates* (Faux and Pratt, 81), as

$$\mathbf{R}_H^{(j)} = \sum \mathbf{x}_H^{(j)} \mathbf{x}_H^{(j)T} \tag{2}$$

where in homogenous coordinates

$$\mathbf{x}_H = \begin{bmatrix} \mathbf{x} \\ \cdots \\ 1 \end{bmatrix}$$

so that now $\boldsymbol{R}_{\mathrm{H}}^{(j)} \in \Re^{(d+1)\times(d+1)}$ and $\boldsymbol{X}^{(j)} \in \Re^{(d+1)\times 1}$. When working in homoheneous coordinates, the constant unit 1 is appended to the vector, so that when the vector is multiplied, it carries lower orders as well. Thus, a single representation encapsulated both first degree (linear) and zero degree (constant) elements. In analogy to the 'correlation matrix memory' and 'autoassociative memory' (see Haykin 1994), the extended matrix $\boldsymbol{R}_{\mathrm{H}}^{(j)}$, in its general form, can be viewed as '*homogeneous autoassociative tensor*'. Now the new matching criterion becomes simply

$$\begin{aligned} i(\mathbf{x}) &= \arg_j \min \left\lfloor \mathbf{x}_H^T \mathbf{R}_H^{(j)^{-1}} \mathbf{x}_H \right\rfloor \\ &= \arg_j \min \left\| \mathbf{R}_H^{(j)^{-\frac{1}{2}}} \mathbf{x}_H \right\| \\ &= \arg_j \min \left\| \mathbf{R}_H^{(j)^{-1}} \mathbf{x}_H \right\|, \quad j = 1,2,\ldots,N \end{aligned} \tag{3}$$

This representation retains the notion of maximum correlation, and for convenience, we now call $\boldsymbol{R}_{\mathrm{H}}^{-1}$ the *synaptic tensor*. Note that this step is based on the fact that the eigenstructure of the extended correlation matrix $\boldsymbol{R}_{\mathrm{H}}^{(j)}$ in homogeneous coordinates corresponds to the principal directions and average (i.e. both second-order and first-order) properties of the cluster accumulated in $\boldsymbol{R}_{\mathrm{H}}^{(j)}$. This property permits extension to higher order tensors, where direct eigenstructure analysis is not well defined. The transition into homogeneous coordinates also dispensed with the need to zero-mean the correlation data.

Digressing for a moment, we recall that the classic neuron possesses a synaptic weight vector $\boldsymbol{w}^{(j)}$ which corresponds to a *point* in the input domain. The synaptic weight $\boldsymbol{w}^{(j)}$ can be seen to be the first order average of its signals, i.e., $\boldsymbol{w}^{(j)}=\Sigma\mathbf{x}_{\mathrm{H}}$ (the last element $\mathbf{x}_{\mathrm{H}\,(d+1)}=1$ of the homogeneous coordinates has no effect in this case). The shape-sensitive neurons hold information regarding the linear correlations among the coordinates of data points represented by the neuron, by using $\boldsymbol{R}_{\mathrm{H}}^{(j)}=\Sigma\ \mathbf{x}_{\mathrm{H}}\mathbf{x}_{\mathrm{H}}^{\mathrm{T}}$. Each element of $\boldsymbol{R}_{\mathrm{H}}$ is thus a proportionality *constant* relating two specific dimensions of the cluster.

Higher Orders

The second-order neuron can be regarded as a second-order approximation of the corresponding data distribution. We may consequently introduce an *m*th*-order* shape sensitive neuron capable of modeling a *d-dimensional* data cluster to an *m*th*-order* approximation. For example, a third-order neuron is capable of storing not only the principal directions and size of the *d*-dimensional data cluster, but also its *curvatures* along each of these axes; hence, it is capable of matching the topology of, say, a Y-shaped fork.

In order to obtain *m*th*-order* components of the data cluster, we use an analogy between eigenstructure decomposition and least-squares fitting. The analogy holds that the principal directions of a data cluster (its eigenvectors) correspond to the normals of the orthogonal set of best-fit hyperplanes through the data set, and the eigenvalues correspond to the variances of the data from those hyperplanes. In homogeneous coordinates, the hyperplanes contain also an *offset*, and hence each homogeneous eigenvector corresponds to the coefficients of the corresponding hyperplane equation. Extending this analogy to higher orders, the higher principal components of the cluster (say, the principal curvatures) correspond to the eigentensors of higher-order correlation tensors in homogeneous coordinates. The neuron is therefore represented by a 2*(m-1)*-dimensional tensor of rank *d+1*, denoted by $\mathbf{Z}^{(j)} \in \Re^{(d+1)\times\ldots\times(d+1)}$. The factor of *2* is introduced by the squared error used by the least-squares method. The tensor is created by successive 'outer products' of the homogeneous vector $\boldsymbol{x}_{\mathrm{H}}$ by itself.

$$\mathbf{Z}_H{}^{(j)} = \mathbf{x}_H{}^{2(m-1)} \tag{4}$$

In practice, in order to extract the winner neuron it is only necessary to determine the amount of correlation between an input and the tensors. As in Eq. (3), in higher orders too this amounts to multiplication of the input $\boldsymbol{x}_{\mathrm{H}}$ by the tensor.

The exponent consists of the factor *(m-1)*, which is the degree of the approximation, and a factor of 2 since we are performing *auto*-correlation so the function i*s* multiplied by itself. In analogy to reasoning that lead to Eq. (3), each eigenvector (now an eigen*tensor* of order *m-1)* corresponds to the coefficients of a principal curve, and multiplying it by an input points produces an approximation of the distance of that input point from the curve. Consequently, the inverse of the tensor $\mathbf{Z}_{\mathrm{H}}^{(j)}$ can then be used to compute the high-order correlation of the signal with the nonlinear shape neuron, by simple tensor multiplication:

$$i(\mathbf{x}) = \arg_j \min \left\| \mathbf{Z}_H^{(j)^{-1}} \otimes x_H^{m-1} \right\|, \quad j = 1,2,\ldots,N \tag{5}$$

where ⊗ denotes tensor multiplication. Note that the covariance tensor can only be inverted if its order is an even number, as satisfied by Eq. (4). Note also that amplitude operator is carried out by computing the root of the sum of the squares of the elements of the argument. The computed metric is now not necessarily spherical and may take various other forms.

In practice however, high-order tensor inversion is not directly required. To make this analysis simpler, we use a Kronecker notation for tensor products (see Graham, 1981). Kronecker tensor product 'flattens out' the elements of $X \otimes Y$ into a large matrix formed by taking all possible products between the elements of *X* and those of *Y*. For example, if *X* is a 2 by 3 matrix, then $X \otimes Y$ is

$$X \otimes Y = \left[\begin{array}{c|c|c} Y \cdot X_{1,1} & Y \cdot X_{1,2} & Y \cdot X_{1,3} \\ \hline Y \cdot X_{2,1} & Y \cdot X_{2,2} & Y \cdot X_{2,3} \end{array}\right] \qquad \textbf{(6)}$$

where each block is a matrix of the size of *Y*. In this notation, the internal structure of higher-order tensors is easier to perceive and their correspondence to linear regression of principal polynomial curves is revealed. Consider, for example, a fourth order covariance tensor of the vector ***x****={x,y}*. The fourth-order tensor corresponds to the simplest non-linear neuron according to Eq. (4), and takes the form of the 2×2×2×2 tensor

$$\langle \mathbf{x} \otimes \mathbf{x} \otimes \mathbf{x} \otimes \mathbf{x} \rangle = \langle \mathbf{R} \otimes \mathbf{R} \rangle = \begin{bmatrix} x^2 & xy \\ xy & y^2 \end{bmatrix} \otimes \begin{bmatrix} x^2 & xy \\ xy & y^2 \end{bmatrix} = \left[\begin{array}{cc|cc} x^4 & x^3y & x^3y & x^2y^2 \\ x^3y & x^2y^2 & x^2y^2 & xy^3 \\ \hline x^3y & x^2y^2 & x^2y^2 & xy^3 \\ x^2y^2 & xy^3 & xy^3 & y^4 \end{array}\right] \qquad \textbf{(7)}$$

The homogeneous version of this tensor includes also all lower-order permutations of the coordinates of $\boldsymbol{x}_H$*={x,y,1}*, namely, the 3×3×3×3 tensor

$$\mathbf{Z}_{H(4)} = \langle \mathbf{x}_H, \mathbf{x}_H, \mathbf{x}_H, \mathbf{x}_H \rangle = \langle \mathbf{R}_H, \mathbf{R}_H \rangle = \begin{bmatrix} x^2 & xy & x \\ xy & y^2 & y \\ x & y & 1 \end{bmatrix} \otimes \begin{bmatrix} x^2 & xy & x \\ xy & y^2 & y \\ x & y & 1 \end{bmatrix} \qquad \textbf{(8)}$$

Extracting Symbolic Rules

It is immediately apparent that the above matrix corresponds to the matrix to be solved for finding a least squares fit of a conic section equation $ax^2+by^2+cxy+dx+ey+f=0$ to the data points. Moreover, the set of eigenvectors of this matrix corresponds to the *coefficients* of the set of mutually orthogonal best-fit conic section curves that are the principal curves of the data. This notion adheres with Gnanadesikan's method for finding principal curves (Gnanadesikan, 1977). Now, substitution of a data point into the equation of a principal curve yields an approximation of the distance of the point from that curve, and the sum of squared distances amounts to the term evaluated in Eq. (5). Note that *each time* we increase complexity, we are seeking a set of principal curves of one degree higher. This implies that the least-squares matrix needs to be *two* degrees higher (because it is minimizing the *squared* error), thus yielding the coefficient *2* in the exponent of Eq. (4) for the general case. Figure 2 shows a cluster of data points and one of their eigentensors.

Rule extraction is performed as follows: First, the synaptic tensor of each neuron is analyzed to extract its eigentensors, which are half the order of the synaptic tensor. The terms of each eigentensor define the coefficients of a polynomial curve, such as the one shown in Figure 2 (b). Distance from this polynomial curve is one of the clustering metrics used by this neuron, and hence can be used as a symbolic rule for classification of the associated data points. The algebraic rule can thus be directly extracted analytically.

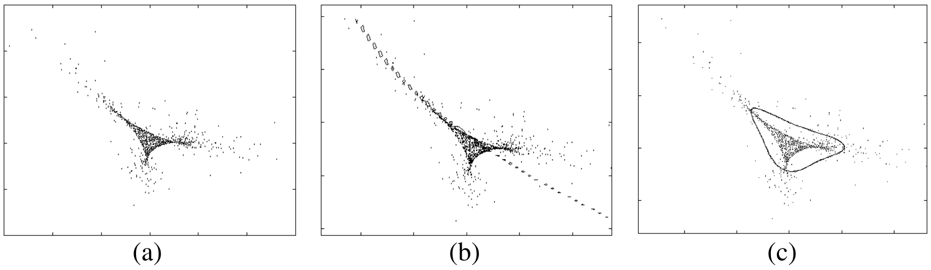

Fig. 2. (a) The cluster of points, (b) one of the six eigentensors, and (c) the best-fit 3rd-order 2D volume corresponding to a unit circle in the space spanned by the six orthogonal eigentensors.

4 Hebbian Learning for High-Order Neurons

In order to show that higher-order shapes are a direct extension of classic neurons, we show that they are also subject to simple Hebbian learning. Following an interpretation of Hebb's postulate of learning, synaptic modification (i.e., learning) occurs when there is a correlation between presynaptic and postsynaptic activities. We have already shown in Eq. (3) above that (a) the presynaptic activity $\boldsymbol{x}_{\mathrm{H}}$ and postsynaptic activity of neuron j coincide when the synapse is strong, i.e. $\boldsymbol{R}_{\mathrm{H}}^{(j)-1}\boldsymbol{x}_{\mathrm{H}}$ is minimum. We now proceed to show that, in accordance with Hebb's postulate of learning, (b) it is sufficient to incur self-organization of the neurons by increasing synapse strength when there is a coincidence of presynaptic and postsynaptic signals.

In order to provide quality (b) above, we need to show how self organization is obtained merely by increasing $\boldsymbol{R}_{\mathrm{H}}^{(j)}$, where j is the winning neuron. As each new data point arrives at a specific neuron j in the net, the synaptic weight of that neuron is adapted by the incremental corresponding to Hebbian learning,

$$\mathbf{Z}_H^{(j)}(k+1) = \mathbf{Z}_H^{(j)}(k) + \eta(k)\mathbf{x}_H^{(j)2(m-1)} \tag{9}$$

where k is the iteration counter and $\eta(k)$ is the iteration-dependent learning rate coefficient. It should be noted that in Eq. 12, $\mathbf{Z}_{\mathrm{H}}^{(j)}$ becomes a *weighted sum* of the input signals $\boldsymbol{x}_{\mathrm{H}}^{2(m-1)}$ (unlike $\boldsymbol{R}_{\mathrm{H}}$ in Eq. 10, which is a *uniform sum*). The eigenstructure analysis of a weighted sum still provides the principal components under the assumption that the weights are uniformly distributed over the cluster signals. This assumption holds true if the process generating the signals of the cluster is stable over time - a basic assumption of all neural network algorithms (Bishop, 1997). In practice this assumption is easily acceptable, as will be demonstrated in the following sections.

In principle, continuous updating of the covariance tensor may create instability in a competitive environment, as a winner neuron becomes more and more dominant. To force competition, the covariance tensor can be normalized using any of a number of factors dependent on the application, such as the number of signals assigned to the neuron so far, or the distribution of data among the neuron (forcing uniformity). Some basic criteria are discussed in (Lipson and Siegelmann, 1999).

5 Implementation

The proposed neuron has been implemented both in an unsupervised and in a supervised setup. High order tensor multiplications have been performed in practice by ‘flattening out’ the tensors using Kroneker’s tensor product notation. Briefly, unsupervised learning is attained by letting randomly initialized neurons compete over input data, using the Hebbian learning and 'winner takes all' principle as described earlier. Supervised learning is attained by training one neuron per input class with inputs of that class only, and then cross-validating the results. The precise implementation is described in (Lipson and Siegelmann, 1999). Below we demonstrate some results.

Figure 3(a) shows three 2nd order neurons (ellipsoids) that self organized to decompose a point set into three natural groups. Note the correct decomposition despite the significant proximity and partial overlap of the cluster, a factor which usually ‘confuses’ classic networks. Figures 3(b,c) show decompositions of point sets using 3rd order neurons, capable of modeling the data with more complex shapes than mere ellipses. Note how the direct determination of the area of overlap permits explicit modeling of uncertainty or ambiguity in the data domain, a factor which is crucial for symbolic understanding. Figure 4 shows three instances of different data topologies, modeled using three 2nd order neurons.

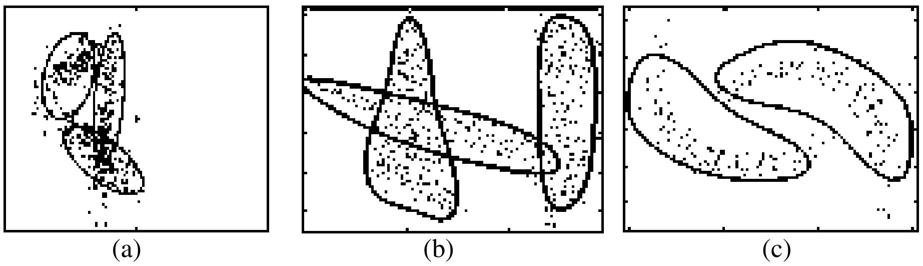

Fig. 3. Self classification of point clusters, (a) 2nd (elliptical), (b,c) 3rd order.

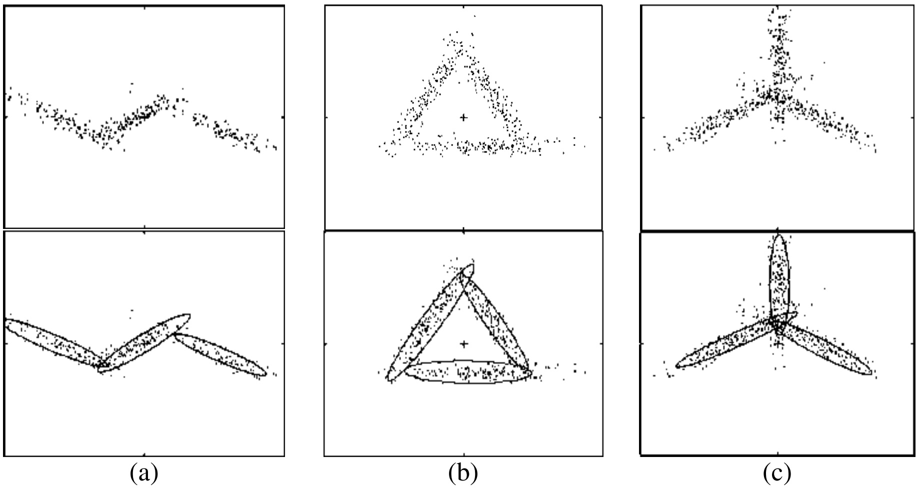

Fig. 4. Point clusters with various topologies and their interpretation, (a) string, (b) hole, (c) fork. Analysis shown was completed after single pass.

The geometric neurons have also been tested on the 4-dimensional IRIS Data benchmark (Anderson, 1939) both in unsupervised and in supervised modes. Out of 150 flowers, the networks misclassified 3 (unsupervised) and 0 flowers (supervised), respectively[1]. Supervised learning was achieved by training each neuron on its designated class separately with 20% cross validation. Tables 1 and 2 summarize the results.

Table 1. Comparison of self-classification results for IRIS data. (Blank cells correspond to unavailable data)

Method	**Epochs**	**# misclassified or unclassified**
Super Paramagnetic (Blatt *et al*, 1996)		25
LVQ / GLVQ (Pal *et al*, 1993)	200	17
K-Means (Mao and Jain, 1996)		16
HEC (Mao and Jain, 1996)		5
2nd-order Unsupervised	20	**4**
3rd-order Unsupervised	30	**3**

Table 2. Supervised classification results for IRIS data. Our results after one training epoch, with 20% cross validation. Averaged results for 250 experiments. (Blank cells correspond to unavailable data)

Order	Epochs	# misclassified	
		Average	Best
3-Hidden-Layer N.N. (Abe *et al*, 1997)	1000	2.2	1
Fuzzy Hyperbox (Abe *et al*, 1997)			2
Fuzzy Ellipsoids (Abe *et al*, 1997)	1000		1
2nd-order Supervised	1	3.08	1
3rd-order Supervised	1	2.27	1
4th-order Supervised	1	1.60	**0**
5th-order Supervised	1	**1.07**	**0**
6th-order Supervised	1	1.20	**0**
7th-order Supervised	1	1.30	**0**

6 Conclusions and Further Research

In this paper, we discussed the use of high-order geometric neurons and demonstrated their practical use for modeling the principal structure of spatial distributions. Although high-order neurons do not directly correspond to neurobiological details, we believe that they can provide powerful symbolic modeling capabilities. In particular, they exhibit useful properties for correctly handling partially overlapping clusters, an occurrence that may represent a key symbolic property. Moreover, when an entire shape or 'rule' is encoded into a single neuron its easy to find a minimal set of 'key examples' that can be used to induce the rule in a learning system. For example, the eigentensors of the synaptic tensor appear to be such a set.

The use of geometric neurons raises some further practical questions, which have not been addressed in this paper, such as selecting the number of neurons for a particular task (network size), the cost implication of the increased number of degrees of freedom, and network initialization. It is also necessary to investigate the relationship between the values of the synaptic tensor and the shapes it may acquire.

Acknowledgements

We thank Eitan Domani for his helpful comments and insight. This work was supported in part by the U.S.-Israel Binational Science Foundation (BSF), by the Israeli ministry of arts and sciences, and by the Fund for Promotion of Research at the Technion. Hod Lipson acknowledges the generous support of the Charles Clore Foundation and the Fischbach Fellowship.

Download further information, MATLAB demos and implementation details of this work from *http://www.cs.brandeis.edu/~lipson/papers/geom.htm*

References

Abe S., Thawonmas R., 1997, "A fuzzy classifier with ellipsoidal regions", *IEEE Trans. On Fuzzy Systems,* Vol. 5, No. 3, pp. 358-368

Anderson E., "The Irises of the Gaspe Peninsula," Bulletin of the American IRIS Society, Vol. 59, pp. 2-5, 1939.

Bishop, C. M., 1997, Neural Networks for Pattern Recognition, Clarendon press, Oxford

Blatt, M., Wiseman, S. and Domany, E., 1996, "Superparamagnetic clustering of data", *Physical Review Letters,* 76/18, pp. 3251-3254

Davé R. N., 1989, "Use of the adaptive fuzzy clustering algorithm to detect lines in digital images", in *Proc. SPIE, Conf. Intell. Robots and Computer Vision*, SPIE Vol. 1192, No. 2, pp. 600-611

Duda R. O., and Hart, P. E., 1973, *Pattern classification and scene analysis*, New York, Wiley.

Faux I. D., Pratt M. J., 1981, *Computational Geometry for Design and Manufacture,* John Wiley & Sons, Chichester

Frigui, H. and Krishnapuram, R., 1996, "A comparison of fuzzy shell-clustering methods for the detection of ellipses", *IEEE Transactions on Fuzzy Systems,* 4/2, pp. 193-199

Geoffrey J. Mclachlan, Thriyambakam Krishnan, 1997, *The EM algorithm and extensions*, Wiley-interscience, New York

Gnanadesikan, R., 1977, Methods for statistical data analysis of multivariate observations, Wiley, New York

Graham A., 1981, Kronecker products and Matrix Calculus: with Applications, Wiley, Chichester

Gustafson E. E. and Kessel W. C., 1979, "Fuzzy clustering with fuzzy covariance matrix", in *Proc. IEEE CDC*, San Diego, CA, pp. 761-766

Haykin, S., 1994, *Neural Networks, A comprehensive foundation*, Prentice Hall, New Jersey

Kavuri, S.N. and Venkatasubramanian, V., 1993, "Using fuzzy clustering with ellipsoidal units in neural networks for robust fault classification", *Computers Chem. Eng.,* 17/8, pp. 765-784

Kohonen, T., 1997,"Self organizing maps", Springer Verlag, Berlin

Krishnapuram, R., Frigui, H. and Nasraoui, O., 1995, "Fuzzy and probabilistic shell clustering algorithms and their application to boundary detection and surface approximation - Parts I and II", *IEEE Transactions on Fuzzy Systems,* 3/1, pp. 29-60.

Lipson H., Siegelmann H. T., 1999, "Clustering Irregular Shapes Using High-Order Neurons", *Neural Computation*, accepted for publication

Mao, J. and Jain, A., 1996, "A self-organizing network for hyperellipsoidal clustering (HEC), *IEEE Transactions on Neural Networks,* 7/1, pp. 16-29.

Pal, N., Bezdek, J.C. and Tsao, E.C.-K., 1993, "Generalized clustering networks and Kohonen's self-organizing scheme", *IEEE Transactions on Neural Networks,* 4/4, pp. 549-557

Pan, J.S., McInnes, F.R. and Jack, M.A., 1996, "Fast clustering algorithms for vector quantization" *Pattern Recognition,* 29:3, pp. 511-518.