A Note on Self-Organizing Semantic Maps

James C. Bezdek, *Fellow, IEEE,* and Nikhil R. Pal, *Member, IEEE*

Abstract—**This paper discusses Kohonen's self-organizing semantic map (SOSM). We show that augmentation and normalization of numerical feature data as recommended for the SOSM is entirely unnecessary to obtain semantic maps that exhibit semantic similarities between objects represented by the data. Visual displays of a small data set of 13 animals based on principal components, Sammon's algorithm, and Kohonen's (unsupervised) self-organizing feature map (SOFM) possess exactly the same qualitative information as the much more complicated SOSM display does.**

I. Introduction

OBJECT data are represented as $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, a set of (n) feature vectors (signals) in feature space $\Re^p$. The jth observed object (some physical entity such as a tank, image, patient, stock market report, etc.) has vector $\mathbf{x}_j$ as its numerical representation; x_{jk} is the kth characteristic (or feature) associated with object j. To characterize feature extraction, let P ($\Re^p$) and P ($\Re^q$) be the sets of all subsets of $\Re^p$ and $\Re^q$, respectively. Let $\Phi: P(\Re^p) \mapsto P(\Re^q)$, be a set-to-set transformation with image $Y = \Phi[X] \in P(\Re^q)$. When $|X| = |Y| = n$, there is a correspondence $\mathbf{x}_i \overset{\Phi}{\leftrightarrow} \mathbf{y}_i \forall i$, and we call Φ a feature extraction transformation [1]. Usually Φ carries points to points, $Y = \{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n\} = \{\Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2), \cdots, \Phi(\mathbf{x}_n)\}$.

The word transformation includes two realizations: Φ may be a function, written $\Phi = f$; or Φ may be an algorithm, written $\Phi = A$. Functions lend themselves to analysis of properties such as linearity, continuity, etc. Algorithms are computational transformations, and hence, their functional properties are generally difficult to verify. We avoid using the word "map" as a synonym for transformation, as there is much confusion in the literature about whether the word is being used in its mathematical sense, its perceptual sense (as a visual display, which is a "map" in a much different sense), or both.

The dimension q can be greater than, equal to, or less than p. Dimensionality is sometimes increased when p is small to enhance the utility of the original data. For example, simple images contain only one dependent variable (intensity) at each pixel in the image. Extracting a list of features such as estimates of the gradient of a picture function and its average intensity over a window centered on each pixel increases the dimensionality of the raw (sensed) data.

When p is large, feature extraction is used for two different but somewhat related problems: dimensionality reduction and visual displays. It is often desirable to reduce p to $q \ll p$. The basic idea is that feature space may be compressed and possibly improved by eliminating, via selection or transformation, redundant (dependent) and unimportant (for the problem at hand) features. If $q \ll p$, time and space complexity of algorithms that use the transformed data are reduced. The simplest method of reduction in dimensionality is feature selection, choosing subsets of the original measured features. Features are selected by taking Φ to be a projection onto some coordinate subspace of $\Re^p$.

The second important use of feature extraction is to get $q = 1$, 2, or three-dimensional scatterplots of X for visual exploratory data analysis. Further, a large class of transformations produce only visual displays from X (and not data sets $Y \subset \Re$, $\Re^2$ or $\Re^3$) through devices other than scatterplots. In this category are functions such as Andrews plots [2]; and algorithms such as Chernoff faces [3], and trees and castles [4]. This more limited class of transformations will be represented as f^D, $A^D: \Re^p \mapsto V(\Re^q)$, and these will be called, respectively, feature display functions and algorithms. The nature of the image space $V(\Re^q)$ of display transformations depends on the function or algorithm being used.

While the primary use of visual displays is in the examination of unlabeled data, feature sets that possess class labels can also be visually examined with profit, for such displays often suggest what type of learning model and/or classifier design can be expected to produce good (or bad) results. Moreover, labeled data can also be used to generate (usually) two-dimensional (2-D) maps exhibiting structural or similarity relations between a set of objects. Any Φ that produces $Y = \Phi[X] \subset \Re^q$ can be used for visual displays by taking $q = 1, 2$, or 3 and plotting Y on a rectangular coordinate system. If the data are labeled, their images have labels in the scatterplot or visual display. These labels can be used to construct visual maps that may indicate similarities between objects, or may be used to design an object data classifier.

The self-organizing feature map (SOFM) is an algorithmic display transformation denoted here by $A^D_{\mathrm{SOFM}} : \Re^p \mapsto V(\Re^q)$ that is implemented through a neural-like network architecture. There are various versions of SOFM that use labeled or unlabeled data. Our interest is in the original algorithm for unlabeled data as described by Kohonen [5] and in a modification of it called the self-organizing semantic map (SOSM) by Ritter and Kohonen [6]. The difference between SOFM and SOSM is that the former produces displays from unlabeled feature vectors in $\Re^p$, while SOSM uses data that are augmented by class labels. The main purpose of our paper is to show that SOFM, principal components, and

Sammon's algorithm all produce the same qualitative results on unaugmented data as SOSM does on the augmented data.

II. Principal Components Analysis and Sammon's Algorithm

Two methods which can be used for both feature extraction and visual display are principal components [7] and Sammon's algorithm [8], [9]. Principal components begins with $S = (1/n)\sum_{k=1}^{n}(\mathbf{x}_k-\mathbf{m})(\mathbf{x}_k-\mathbf{m})^T$, the sample covariance matrix of X, where $\mathbf{m} = (1/n)\sum_{k=1}^{n}\mathbf{x}_k$ is the sample mean of X. Assuming S to be positive definite, extract and order the p eigenvalues of S, as, say, $\lambda_1 \geq \lambda_2 \geq \cdots \lambda_p > 0$, and let $\mathbf{v}_i$. $i = 1, 2, \cdots, p$ be the corresponding unit eigenvectors; i.e., $S\mathbf{v}_i = \lambda_i\mathbf{v}_i, i = 1, 2, \cdots, p$ and $\mathbf{v}_i^T\mathbf{v}_i = 1 \forall i = 1, 2, \cdots, p$. Eigenvector $\mathbf{v}_i$ is called the ith principal vector of S. The set $\{\mathbf{v}_i\}$ is an orthonormal basis for $\Re^p$ which is simply a rotation of the canonical basis that decorrelates the samples.

The principal vectors of S are used to define p linear feature extraction functions. Let P_q denote the $p \times q$ matrix whose q columns are the first q (ordered) eigenvectors of S. Thus

$$P_1 = \begin{bmatrix} | \\ \mathbf{v}_1 \\ | \end{bmatrix}, P_2 = \begin{bmatrix} | & | \\ \mathbf{v}_1 & \mathbf{v}_2 \\ | & | \end{bmatrix}, \cdots,$$

$$P_q = \begin{bmatrix} | & | & & | \\ \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_q \\ | & | & & | \end{bmatrix}.$$

Now define, for $q = 1, 2, \cdots, p$, $f_{PC,q}: \Re^p \mapsto \Re^q$ as $\mathbf{y}_q = f_{PC,q}(\mathbf{x}) = P_q^T\mathbf{x}$. $\mathbf{y}_q$ is called the qth order principal component of $\mathbf{x}$, and $y_{qi} = \mathbf{v}_i^T\mathbf{x}$ is called the ith score of $\mathbf{x}$. In particular, the second-order PC image of $\mathbf{x}$, $f_{PC,2}(\mathbf{x})$ is a point in $\Re^2$, and a plot of the set $Y_{PC,2} = \{f_{PC,2}(\mathbf{x}_i): \mathbf{x}_i \in X\}$ is called a principal component scatterplot of the first and second principal components of X. Of course any of the $p(p-1)/2$ pairs of PC's of the data may be plotted; of these, our notation accounts only for the single pair produced by $f_{PC,2}$. For example, a scatterplot of the last two PC's is sometimes examined to see what is in the "tail" of the sample variance.

For any $q \leq p$, $Y_{PC,q} = f_{PC,q}[X]$ is data in $\Re^q$ extracted from X, and this extraction produces an image set with a known statistical property. The projection of X onto the span of $\{\mathbf{v}_1, \mathbf{v}_2, \cdots \mathbf{v}_q\}$ accounts for the maximum possible fraction of the total sample variance in X that can be accounted for by a linear projection onto a q-dimensional vector subspace, and this is measured directly by the ratio $E_{PC,q}(Y) = (\sum_{i=1}^{q}\lambda_i)/(\sum_{i=1}^{p}\lambda_i)$. In a rough sense then, plotting the first few principal components allows you to visually examine those features in the data that account for most of its variance.

Sammon's method is a set-to-set algorithm denoted here by $A_{S,q}: P(\Re^p) \mapsto P(\Re^q)$. $A_{S,q}$ attempts to find a set $Y_{S,q} = A_{S,q}[X]$ in $\Re^q$ such that the distances between pairs of vectors in X are preserved in their algorithmic images in $Y_{S,q}$. Let d_{ij}^*, d_{ij} be, respectively, the Euclidean distances between $\mathbf{x}_i$, $\mathbf{x}_j$ in $\Re^p$, and (unknown) vectors, $\mathbf{y}_i$, $\mathbf{y}_j$ produced by $A_{S,q}$ in $\Re^q$. Sammon suggested looking for $Y_{S,q}$ by minimization of an error function $E_{S,q}$ that measures how well the configuration of the data points in $Y_{S,q}$ agree with those in data set X in the sense of matching pairwise distances. Sammon's measure of error is

$$E_{S,q}(Y) = \frac{1}{\sum_{i<j} d_{ij}^*} \sum_{i<j} \frac{(d_{ij}^* - d_{ij})^2}{d_{ij}^*}. \tag{1}$$

$E_{S,q}(Y) = 0$ if and only if $A_{S,q}$ preserves all $n(n-1)/2$ pairs of distances exactly. Thus, $A_{S,q}$ attempts to be an isometric connector between X and $Y_{S,q} = A_{S,q}[X]$. Minimization of $E_{S,q}(Y)$ is an unconstrained optimization problem in the nq variables y_{ij}, $i = 1, 2, \cdots, n$; $j = 1, 2, \cdots, q$. Sammon's algorithm is the method of steepest descent for (approximate) minimization of $E_{S,q}(Y)$. Modifications of Sammon's algorithm to improve its speed and performance are discussed by Schachter [10], Pykett [11], Chang and Lee [12], and Biswas *et al.* [13].

To summarize, principal components $[E_{PC,q}(Y)]$ and Sammon's algorithm $[E_{S,q}(Y)]$ are both driven by well-defined performance criteria that attempt to guide them toward extracted features that satisfy a desirable property. Neither of these algorithms requires class label information, and they produce extracted feature vectors in $\Re^q$ for any $q \leq p$. At the termination of either algorithm, the vectors $Y_{PC,2}$ or $Y_{S,2}$ can be scatterplotted on a standard rectangular coordinate system, and these images of X can be inspected to guess about its substructure. We will show that these two algorithms produce—without the benefit of any label (or linguistic) information—images that are qualitatively identical to those produced by the SOSM, which, according to Ritter and Kohonen, must include the label information to display semantic relations between symbolic data [12, p. 241].

III. The Self-Organizing Feature Map

The SOFM is an algorithmic display transformation denoted here by $A_{\text{SOFM}}^D : \Re^p \mapsto V(\Re^q)$ that is often advocated for visualization of metric-topological relationships and distributional density properties of feature vectors (signals) X in $\Re^p$ [5]. SOFM is implemented through a neural-like network architecture that is believed to be similar in some ways to the biological neural network. Fig. 1 illustrates this architecture for $q = 2$.

The visual display produced by A_{SOFM}^D presumably helps one form hypotheses about topological structures in X. In principle X can be transformed onto a display lattice in $\Re^q$ for any q; in practice, visual displays can be made only for $q \leq 3$ and are usually made on a linear or planar configuration arranged as a rectangular or hexagonal lattice. In this article we concentrate on square $(m \times m)$ displays in $\Re^2$.

Input vectors $\mathbf{x} \in \Re^p$ are distributed by a fan-out layer to each of the $(m \times m)$ output nodes in the competitive layer. Each node in this layer has a weight vector prototype $\mathbf{v}_{ij}$ attached to it as shown in Fig. 1. We let $O_p = \{\mathbf{v}_{ij}\} \subset \Re^p$ denote the set of m^2 weight vectors. O_p is (logically) connected to a display grid $O_2 \subset V(\Re^2)$. (i, j) in the index

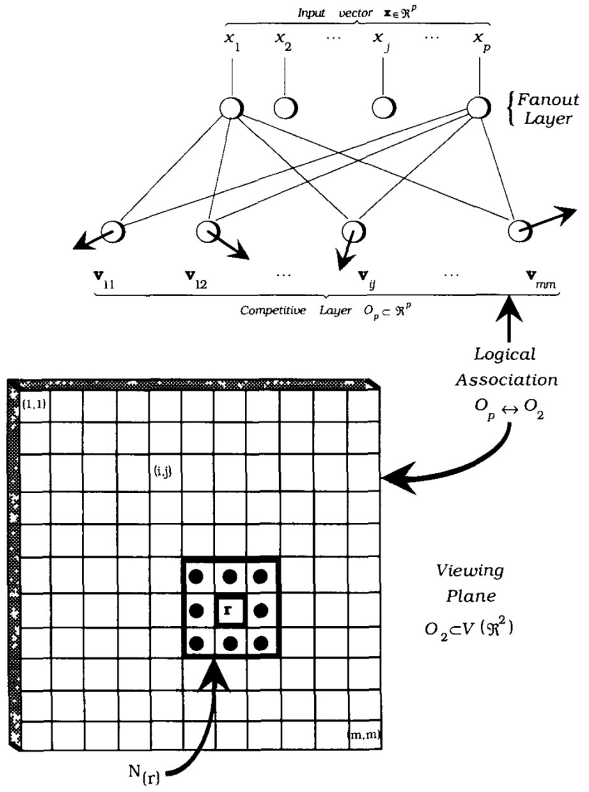

Input vector $\mathbf{x} \in \Re^p$
x_1 x_2 ... x_j ... x_p
Fanout Layer
$\mathbf{v}_{11}$ $\mathbf{v}_{12}$ ... $\mathbf{v}_{ij}$ ... $\mathbf{v}_{mn}$
Competitive Layer $O_p \subset \Re^p$
Logical Association
$O_p \leftrightarrow O_2$
(1,1)
(i,j)
r
m,m
Viewing Plane
$O_2 \subset V(\Re^2)$
N(r)

Fig. 1. The SOFM architecture with iterate number (t) suppressed.

set $\{1, 2, \cdots, m\} \times \{1, 2, \cdots, m\}$ is the logical address of the cell. There is no vector with coordinates in $\Re^2$ associated with each cell. This gives a one-to-one correspondence between the m^2 p-vectors $\{\mathbf{v}_{ij}\}$ and the m^2 cells $\{(i, j)\}$, i.e., $O_p \leftrightarrow O_2$. In the literature display cells are sometimes called nodes, or even neurons, in deference to possible biological analogs.

SOFM begins with a (usually) random initialization of the weight vectors $\{\mathbf{v}_{ij}\}$. Having made the important point that feature vectors $\mathbf{x}$ in $\Re^p$ are logically identified with 2-D addresses in O_2 as in Fig. 1, we now simplify our notation by suppressing double subscripts. Now let $\mathbf{x} \in \Re^p$ enter the network and let t denote the current iterate number. Find $\mathbf{v}_{r,t-1}$, the vector in O_p that best matches $\mathbf{x}$ in the sense of minimum Euclidean distance in $\Re^p$. This vector has a (logical) "image" which is the cell in O_2 with subscript r. Next, a topological (spatial) neighborhood $N_t(r)$ centered at r is defined in O_2, and its display cell neighbors are located. In Fig. 1 we show a 3×3 window $N(r)$ centered at r; this would correspond to updating nine prototypes in $\Re^p$. Finally, $\mathbf{v}_{r,t-1}$ and the other weight vectors associated with cells in the spatial neighborhood $N_t(r)$ are updated using the rule

$$\mathbf{v}_{i,t} = \mathbf{v}_{i,t-1} + h_{ri}(t)(\mathbf{x} - \mathbf{v}_{i,t-1}). \qquad (2)$$

Here r is the index of the "winner" prototype

$$r = \underbrace{\arg\min}_{i} \{\|\mathbf{x} - \mathbf{v}_{i,t-1}\|\} \qquad (3)$$

and $\|*\|$ is the Euclidean norm on $\Re^p$ (this is the only norm we use in this paper). The function $h_{ri}(t)$ is used to express the strength of interaction between cells r and i in O_2. Usually $h_{ri}(t)$ decreases with t, and for fixed t it decreases as the distance (in O_2) from cell i to cell r increases. A common choice for $h_{ri}(t)$ is $h_{ri}(t) = \alpha_t e^{-dist^2(r,i)/\sigma_t^2}$, where α_t and σ_t decrease with time t. The topological neighborhood $N_t(r)$ also decreases with time. Note that the "preimages" of $N_t(r)$ are not necessarily metrical neighbors in R^p. This scheme, however, often preserves spatial order in the sense that weight vectors which are metrically close in $\Re^p$ generally have, at termination of the learning procedure, visually close images in the viewing plane.

There are many variations of the basic SOFM algorithm. For example, in [14] an algorithm is suggested that uses metrically defined neighborhoods of winners $\mathbf{v}_r$ in feature space $\Re^p$. In [15] neighborhoods of winner $\mathbf{v}_r$ in $\Re^p$ are chosen from nodes in a minimal spanning tree constructed on the weight vector set $O_p = \{\mathbf{v}_{ij}\} \subset \Re^p$. Here is the version of the SOFM that is used in the numerical example given later.

Algorithm A^D_{SOFM} (Kohonen [5]):

```
Begin
  Input $X$              /** unlabeled data set
                          $X = \{\mathbf{x}_i \in \Re^p : i = 1, 2, \cdots, n\}$ **/
  Input $m$              /** the display grid size, a square
                          $m \times m$ lattice is assumed **/
  Input $maxstep$        /** maximum number of
                          updating steps **/
  Input $N_0$            /** initial neighborhood size **/
  Input $\alpha_0$       /** the initial step size
                          (learning coefficient) **/
  Input $\sigma_0$ and $\sigma_f$   /** parameters to control
                          effective step size **/
/** Learning phase **/
  Randomly generate initial weight vectors
    $\{\mathbf{v}_{ij}, i = 1, 2, \cdots, m; j = 1, 2, \cdots, m\}$
  $t \leftarrow 0$
  While ($t < maxstep$)
    Select randomly $\mathbf{x}(t)$ from $X$;
    find $r = \underbrace{\arg\min}_{i} \{\|\mathbf{x}(t) - \mathbf{v}_i(t)\|\}$ /** $r$ and $i$
      stand for two- dimensional indexes that uniquely
      identify a weight vector in $O_p$ **/
    $\mathbf{v}_i(t+1) \leftarrow \mathbf{v}_i(t) + \alpha_t g_t[dist(r, i)][\mathbf{x}(t) - \mathbf{v}_i(t)]$
      $\forall i \in N_t(r)$
    $\mathbf{v}_i(t+1) \leftarrow \mathbf{v}_i(t) \forall i \notin N_t(r)$ /** $dist(r, i)$
      is the Euclidean distance between centers of
      nodes $r$ and $i$ on the display lattice,
      $g_t(d) = e^{-d^2/\sigma_t^2}$ **/
    $t \leftarrow t + 1$
    $\alpha_t \leftarrow \alpha_0(1 - t/maxstep)$
    $N_t \leftarrow N_0 - t(N_0 - 1)/maxstep$
    $\sigma_t \leftarrow \sigma_0 - t(\sigma_o - \sigma_f)/maxstep$
    /** - there are many other ways to readjust
    $\alpha_t$, $N_t$, and $\sigma_t$,
    and many choices for $g_t$ **/
  While End
```

```
/** Display phase**/
  For each $\mathbf{x} \in X$ find $r = \underbrace{\arg\min}_{i} \{\|\mathbf{x} - \mathbf{v}_i\|\}$,
    and mark the associated cell $r$ in $O_2$.
End.
```

According to Kohonen [5], there are two opposing tendencies at work in the self-organizing process. First, the weight vectors tend to describe the density function of the input population, and second, the local interaction between processing units tends to preserve continuity in the double (2-D) sequences of weight vectors. In other words, the weight vectors in O_p are trying simultaneously to approximate the distribution of the data in $\Re^p$ and to have logical images which are topologically ordered in O_2. At the conclusion of learning a final pass is made through X to get the display in $V(\Re^2)$, which is produced by "lighting up" (marking) each cell r in O_2 that corresponds to a winner node $\mathbf{v}_r \in O_p$.

The SOFM in its original form is an unsupervised learning algorithm. Ritter and Kohonen suggested that the SOFM could not be used to discover semantic relationships without "an essential, new ingredient [which] is the inclusion of the contexts, in which each symbol appears, into the input data" [6, p. 241]. They used the class information or label vector for each point in feature data to produce what they called a semantic map, which exhibits the similarity relationship between a set of animals. Mitra and Pal [16] have also used class information in a similar manner to design a fuzzy self-organizing classifier. These authors augment the feature vectors (attribute vectors) with the class information and use the augmented data as input to the network. The primary objectives of this paper are to establish that using class information as discussed in [6] and [16] is counter-intuitive and that qualitatively identical results can be obtained without using class information. We will establish this by showing that feature extraction techniques such as Sammon's algorithm and principal components and the SOFM display method all generate qualitatively identical results to the SOSM.

IV. The Self-Organizing Semantic Map

The method we describe next is the SOSM [6].[1] The basic idea is to encode class information of objects as a part of the input data. This is done as follows: Let $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, be a set of (n) feature vectors (signals) in feature space $\Re^p$ that possesses representatives of each of c known classes. For each $\mathbf{x}_i$ there is output information, which in its simplest form is the crisp (nonfuzzy) label of the class to which $\mathbf{x}_i$ belongs.

Ritter and Kohonen [6] suggested that direct application of A^D_{SOFM} to object data without class labels would not yield results that displayed semantic relationships or similarities between the objects. Accordingly, SOSM begins by changing the input data set X to

$$\hat{X} = \left\{ \hat{\mathbf{x}}_i = \begin{bmatrix} \alpha \mathbf{x}_{s,\,i} \\ \mathbf{x}_{a,\,i} \end{bmatrix} \right\} \in \Re^{c+p}$$

where $\mathbf{x}_{a,\,i}$ is the original attribute vector $\mathbf{x}_i \in \Re^p$ and $\mathbf{x}_{s,\,i}$ contains the class information. Following Ritter and Kohonen [6], we shall call $\mathbf{x}_{s,\,i}$ the symbol part of each object (modulated by $\alpha \in (0, 1]$). The purpose of the multiplier α is to make sure that during learning the class information does not get much importance compared to the attribute part. For the simplest case, $\mathbf{x}_{s,\,i} = \{0, 0, \cdots, \underbrace{1}_{k}, 0, \cdots, 0\}^T \in \Re^c$ when $\mathbf{x}_i$ is in class k. Often $\hat{\mathbf{x}}_i$ is normalized to unit length. The network is then trained with the normalized version of $\hat{X}$, and after training some or all of the display cells (nodes) are marked (labeled) by the class (symbol) information. The marking strategy can vary with the goal at hand. Ritter and Kohonen used SOSM to generate what they called a semantic map of an animal data set, X_A = Animal. X_A describes each of 16 animals by a set of 13 binary attributes,[2] as listed in Table I.

TABLE I
X_A Animal (Ritter and Kohonen [6])

	Animal ↦	dove	hen	duck	goose	owl	hawk	eagle	fox	dog	wolf	cat	tiger	lion	horse	zebra	cow
	small	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0
Is	medium	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
	big	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
	2 legs	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
	4 legs	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
Has	hair	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
	hooves	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
	mane	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	0
	feathers	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
	hunt	0	0	0	0	1	1	1	1	0	1	1	1	1	0	0	0
Likes	run	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0
to	fly	1	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0
	swim	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

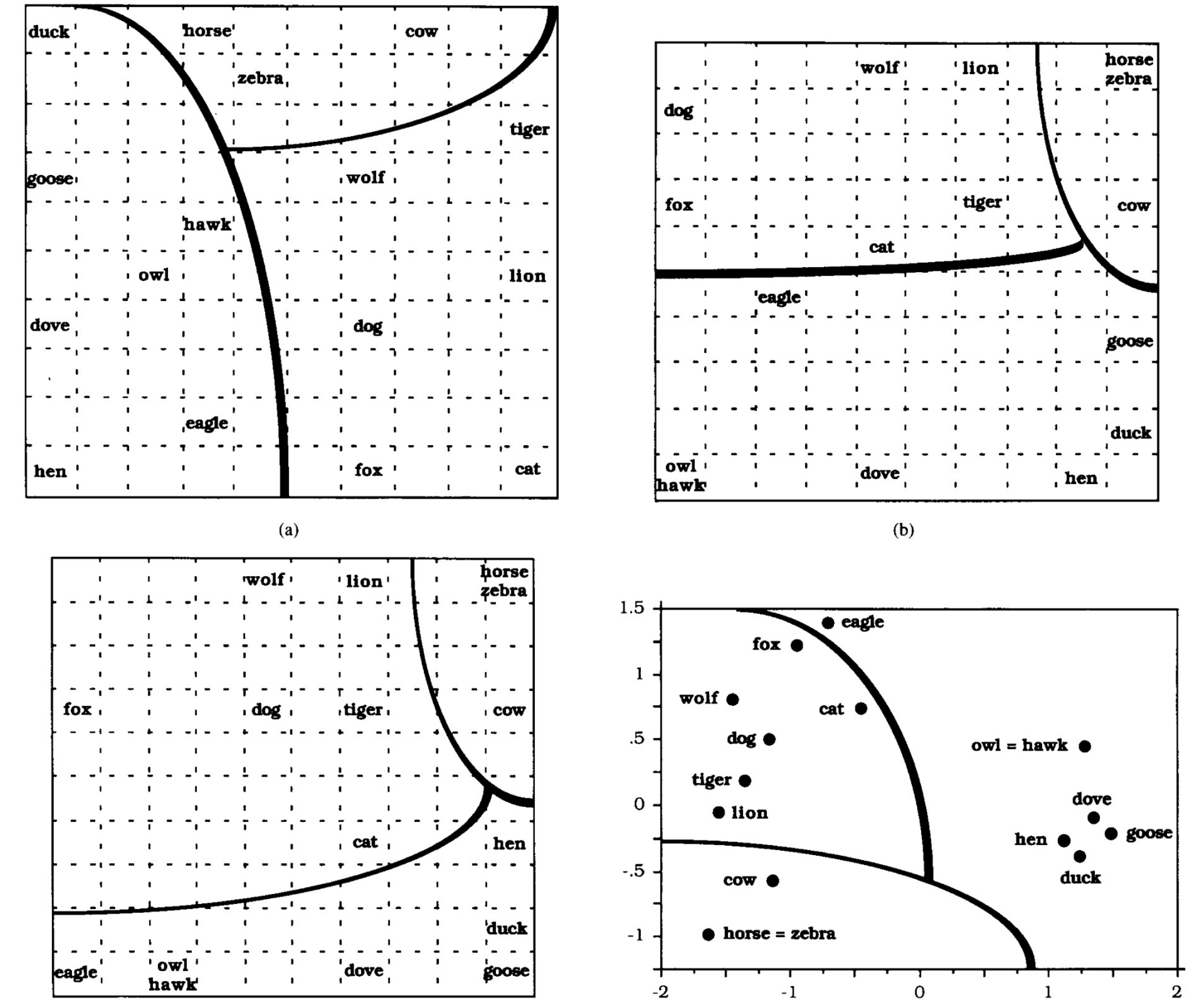
For this data set the number α multiplying $\mathbf{x}_{s,\,i}$ marks (encodes) attribute vector $\mathbf{x}_{a,\,i}$ as the ith animal as only the ith place in $\mathbf{x}_{s,\,i}$ contains a nonzero value. Normalizing $\mathbf{x}_i = [{}^{\alpha \mathbf{x}_{s,\,i}}_{\mathbf{x}_{a,\,i}}]$ to have unit length and using $\alpha = 0.2$ as the marker value (the choice of α clearly affects the results, and there is no guidance about how to select it) were recommended and discussed in [6, p. 247].

Quote A: "The core idea underlying symbol maps is that the two parts are weighted properly such that the norm of the attribute part predominates over that of the symbol part during the self-organizing process; the topographical mapping then mainly reflects the metric relationships of the attribute sets."

Quote B: "However, we now want to be sure that the encoding of the symbols does not convey any information about similarities between the items."

[1] Authors such as Mitra and Pal [16] do not use this terminology. We prefer to stick with the original name given to this algorithm by Ritter and Kohonen.

[2] Ritter and Kohonen refer in [6, p. 241] to these 13-dimensional attribute vectors as the context (of the data) in their demonstration 1.



(a)

(b)

(c)

(d)

Fig. 2. (a) The SOSM display for X_A = Animal: normalize and augmented (after Fig. 2, [6]). (b) A^D_{SOFM} display for unaugmented X_A = Animal: normalized. (c) A^D_{SOFM} display for unaugmented X_A = Animal: unnormalized. (d) $f_{PC,2}$ scatterplot display for X_A = Animal: unnormalized.

Quotes A and B seem on the one hand to weaken the authors resolve to use the symbol part that they recommend adding to the data. On the other hand, we read elsewhere that:

Quote C: "If then, during recognition of input information, the attribute signals are missing or are weaker, the (same) map units are selected on the basis of the symbol part solely."

Quote C seems counter to Quotes A and B, suggesting that the symbol part is very important indeed. We repeat these statements to emphasize several difficulties with the method advocated in [6]. First, there is the problem of choosing proper weights. Second, for, say, 1000 objects (animals in this case) the augmented data lie in $\Re^{1013}$, so distance computations may be excessive, even though each symbol vector might contain 999 zeroes. And while the third remark above is offered as justification for the symbol vectors, we wonder why they are needed at all? If we have the symbol part, then the data are labeled and there is no need to classify the objects. Given the symbol information, you can even get the characteristics (description) of the corresponding animal (or meaning of the symbol) by extraction of the weight vector associated with that symbol.

It is true that for recognition of the objects (animals here) or calibration of a classifier, the symbol (or class) information must be incorporated into the SOFM visual display. We claim this can be easily done without recourse to augmentation of the attribute data (the context in [6]) with symbol information. The SOFM display can be (and should be, as implied by quotations A and B) obtained without the symbol information. Once the SOFM display is formed (i.e., learning is over) the symbol information can be simply attached to the winner cell corresponding to each animal. The resulting display, called a semantic map by Ritter and Kohonen, will (or at least could) show semantic relationships between the symbolic data as expressed by their attribute vector representations.

This will ensure what has been demanded by Quotes A and B.

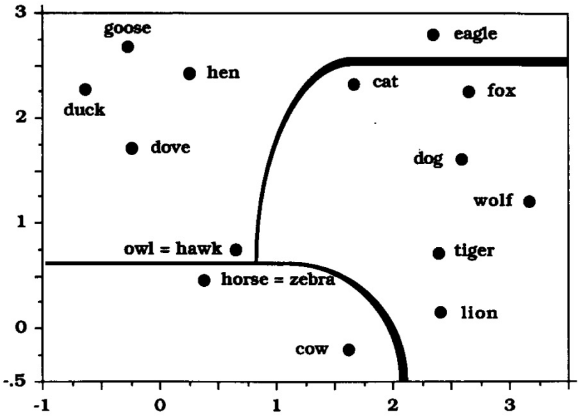
If it is desired to now label the rest of the display cells (and their preimages, the other prototypes in $\Re^p$), perhaps to equip the network as a nearest prototype classifier over the range of the input space, this can also be done without recourse to the SOSM. First, we apply SOFM as described earlier without the label information to the attribute vectors. Now for each node (weight vector) in O_p, find the activation values for all the attribute vectors in the training data. Suppose for the ith node the maximum response (activation/match) is obtained for the jth object (animal here); then mark the ith node (in O_p and O_2) as the jth object. This procedure will assign a label to each of the m^2 nodes in the SOFM map.[3] This map corresponds to the "simulated electrode penetration mapping" in Fig. 3 of Ritter and Kohonen [6, p. 248].

Once this is done, given any new unlabeled attribute vector, we can feed it to the network and find the unit that produces the maximum response (match). The label of the best matching unit is then assigned to the unlabeled attribute vector under consideration. This converts the SOFM display into a nearest prototype classifier. Again, there is no need to use $\hat{X} = \{\hat{\mathbf{x}}_i = \left[\begin{smallmatrix}\alpha \mathbf{x}_{s,i} \\ \mathbf{x}_{a,i}\end{smallmatrix}\right]\} \in \Re^{c+p}$ with the SOSM to achieve maps like Fig. 3 in Ritter and Kohonen; it is easily done with the SOFM if labels for the original objects are known. Each object that has distinct attribute coordinates will acquire a region in the display grid with an area of at least one cell.

V. The Numerical Example

Fig. 2(a) (Fig. 2 in [6]) is the result obtained by Ritter and Kohonen upon applying their SOSM algorithm to the normalized, augmented data. Notice that horse and zebra occupy distinct cells in the SOSM display, even though their attribute vectors are identical. The same is true for owl and hawk. The boundaries in this and all subsequent views of X_A are hand drawn to emphasize that contiguously placed animals could be logically (i.e., semantically) grouped. The boundaries in Fig. 2(a) show that three subgroups of animals that seem by human intuition (or knowledge) to be closely related do occupy subregions of the SOSM display.

For unnormalized augmented data the semantic relationship is governed by the metric relationship between the attribute vectors because $||\mathbf{x}_{s,i} - \mathbf{x}_{s,j}||^2 = 2 \quad \forall i, j; \quad i \neq j$. Thus the metric relation between the augmented vectors do not change due to the incorporation of the symbol part for unnormalized data. Two animals should be different because their characteristics are different. Since a horse is clearly not a zebra, the defect from the point of view of discrimination lies with an improper choice of numerical attributes. In the animal data set horse and zebra have the same attribute vectors without the symbol part and so SOFM (but not SOSM) will not and should not be able to discriminate between them. It is not desirable to have different positions on the display for the same feature vector, as occurs in Fig. 2(a). This is not a limitation of the display but rather of the inadequacy of the chosen set of attributes. This can be avoided with a different set of numerical attributes; it should not be avoided by attaching label information to each attribute vector.



(e)

Fig. 2. Continued. (e) $A_{S,2}$scatterplot display for X_A = Animal: unnormalized.

To appreciate this problem, consider a simple example. Let

$$\mathbf{x} = \begin{bmatrix}\mathbf{x}_s \\ \mathbf{x}_a\end{bmatrix} = \begin{bmatrix}\mathbf{x}_s \\ 0\end{bmatrix} + \begin{bmatrix}0 \\ \mathbf{x}_a\end{bmatrix}$$

be the vectorial representation of an object (an animal in this case), where $\mathbf{x}_s$ is a d-dimensional vector representing the symbol part of the data and $\mathbf{x}_a$ is the attribute vector in $\Re^p$. We think that Quote B means that Ritter and Kohonen want, for every $i \neq j$, $||\mathbf{x}_{s,i} - \mathbf{x}_{s,j}|| = c$, where c is a constant and $\mathbf{x}_{s,i}$, $\mathbf{x}_{s,j}$ are the symbol vectors encoding the ith and jth animal labels, respectively. For the Euclidean norm, for example, $c = \alpha\sqrt{2}$. As long as the data are not normalized the condition $||\mathbf{x}_{s,i} - \mathbf{x}_{s,j}|| = c$ will hold. What happens when $\mathbf{x}$ is normalized?

For clarity we explain this with only three animals that each possess three binary features. So $\mathbf{x}_{s,i}$ and $\mathbf{x}_{s,j}$ will each have three components. Let the three augmented vectors be

$$\begin{pmatrix}\alpha \\ 0 \\ 0 \\ 1 \\ 1 \\ 1\end{pmatrix}, \quad \begin{pmatrix}0 \\ \alpha \\ 0 \\ 1 \\ 0 \\ 1\end{pmatrix}, \quad \begin{pmatrix}0 \\ 0 \\ \alpha \\ 0 \\ 0 \\ 1\end{pmatrix},$$

$$\mathbf{x}_{s,1} = \begin{pmatrix}\alpha \\ 0 \\ 0\end{pmatrix}, \quad \mathbf{x}_{s,2} = \begin{pmatrix}0 \\ \alpha \\ 0\end{pmatrix}, \quad \mathbf{x}_{s,3} = \begin{pmatrix}0 \\ 0 \\ \alpha\end{pmatrix}.$$

Prior to normalization we have $||\mathbf{x}_{s,1} - \mathbf{x}_{s,2}|| = ||\mathbf{x}_{s,1} - \mathbf{x}_{s,3}|| = ||\mathbf{x}_{s,2} - \mathbf{x}_{s,3}|| = \alpha\sqrt{2}$, and that is what is desired. Normalization leads to

$$\mathbf{x}_{s,1} = \begin{pmatrix}\alpha/\sqrt{3+\alpha^2} \\ 0 \\ 0\end{pmatrix},$$

[3] The map produced this way using SOFM on the numerical attributes alone will differ from Fig. 3 of [6] in that it will have one region named "owl-hawk" and one named "horse-zebra" rather than separate regions for each of these four animals.

$$\mathbf{x}_{s,2} = \begin{pmatrix} 0 \\ \alpha/\sqrt{2+\alpha^2} \\ 0 \end{pmatrix},$$

and

$$\mathbf{x}_{s,3} = \begin{pmatrix} 0 \\ 0 \\ \alpha/\sqrt{1+\alpha^2} \end{pmatrix}.$$

Now $||\mathbf{x}_{s,1} - \mathbf{x}_{s,2}|| = ||\mathbf{x}_{s,1} - \mathbf{x}_{s,3}|| = ||\mathbf{x}_{s,2} - \mathbf{x}_{s,3}|| = c$ is no longer valid. After normalization, the symbol part of each vector contributes to the proximity relations between the three animals. For example, suppose $\alpha = 0.2$. Then $||\mathbf{x}_{s,1} - \mathbf{x}_{s,2}|| = 0.181 < ||\mathbf{x}_{s,1} - \mathbf{x}_{s,3}|| = 0.227 < ||\mathbf{x}_{s,2} - \mathbf{x}_{s,3}|| = 0.240$. This is a proximity relation that Ritter and Kohonen did not intend to have (cf., Quote B). The only effect of this representation and normalization is to have two different positions on the SOSM map for two objects with identical attribute (or context) vectors. In other words, when the attribute vectors say that two objects are identical, symbolic augmentation forces them to occupy different positions in the map for an unnatural reason. This is not sensible or rational; it simply means that if the animals are really different (and they are, of course), the numerical representation chosen is inadequate to reflect this.

To justify our claims, we now show the semantic maps produced by plotting $Y_{PC,2}$, $Y_{S,2}$, and the visual displays made by both SOFM and SOSM when applied to X_A. We shall also illustrate that normalization of the features is not necessary. Here are the computational protocols we used: Sammon's algorithm was initialized randomly and run for 300 iterations with $\alpha = 0.35$. A^D_{SOFM} used $\alpha_0 = 0.9$; $\sigma_0 = 4.0$; $\sigma_f = 0.5$; $N_0 = 9$; and $m = 10$. A^D_{SOFM} was terminated following Kohonen's suggestion [17], by using maxstep $= 500 * m^2 = 500 * 10 * 10 = 50\,000)$ steps. For a fixed data set one can cycle A^D_{SOFM} sequentially through X, but in this study randomly selected data points were used at each t.

We applied A^D_{SOFM} to the unaugmented data X_A of Table I in both normalized and unnormalized forms. Fig. 2(b) is the result with normalized data. Fig. 2(c) is the display obtained using unnormalized data. A few of the animals occupy different spatial positions in Fig. 2(b) and (c), but the qualitative information they possess (i.e., that contiguous regions of the display contain the three families) is the same and is again the same as reported in [6]. One of the main points of [6] was to show that the data needed to be modified before A^D_{SOFM} could be successful with data that have "semantic" content. The boundaries we have drawn in Fig. 2(b) and (c) indicate the same "family relationships" sought and found in [6] and exhibited in Fig. 2(a). Comparing Fig. 2(b) and (c) to Fig. 2(a) shows that the normalization and symbolic augmentation procedures recommended in [6] are both unnecessary for this data. Kohonen's A^D_{SOFM} does just fine.

Fig. 2(d) and (e) are scatterplot displays of the 2-D feature vectors extracted from the original, unnormalized 13-dimensional data set X_A shown in Table I with principal components $f_{PC,2}$ and Sammon's $A_{S,2}$, respectively. As you can see, boundaries can be drawn on each display that isolate the three groups of animals exactly as in Fig. 2(a)–(c). The various animals occupy different regions of $\Re^2$ in these views, but the qualitative information contained by these outputs is identical to that in Fig. 2(a)–(c). Since owl and hawk and horse and zebra have the same attribute vector representations (columns (5, 6) and (14, 15) in Table I), they are mapped to the same vectors by both extraction methods. In [6], the normalization of the symbol part of each datum for these two pairs of animals causes their images to occupy different cells as seen in Fig. 2(a), which seems artificial at best—after all, these animals have identical attribute (context) vectors. Columns (2, 3) = "hen" and "duck" and (1, 4) = "goose and dove" differ only in one coordinate. Consequently, these animals have distinct vector images under all three methods discussed here.

Displays analogous to Fig. 2(d) and (e) were made by processing the normalized version of X_A that was used for Fig. 2(b) with principal components and Sammon's algorithm, and the qualitative results were still the same. Thus, the qualitative information possessed by Fig. 2(a) is easily obtained directly from the original data in a straightforward manner using SOFM, principal components, or Sammon's algorithm.

VI. Conclusions

There is nothing extraordinary about the data set X_A = Animal. Symbolic augmentation and normalization as discussed in [6] are not required to secure good qualitative results that display semantic relationships between data that have class labels. Kohonen's SOFM algorithm works quite nicely on the unnormalized, unaugmented feature vector data X_A and yields exactly the same family relationships as principal components and Sammon's algorithm. Considering the simplicity of these latter algorithms, we wonder whether the SOSM is really a value-added deviation from the original SOFM.

One further comment. The idea of augmenting the original data was carried a step further by Mitra and Pal [16], who borrowed Ritter and Kohonen's idea to design a classifier. Their fuzzy SOFM network can be viewed as a three-step process: learning, calibration, and testing. For learning, a data point $\mathbf{x} \in \Re^p$ is replaced (not augmented) by a point $\hat{\mathbf{x}} \in \Re^{3p+c}$; $\hat{\mathbf{x}} = [\mathbf{x}', \alpha\mathbf{x}'']^T, \mathbf{x}' \in \Re^{3p}$, and $\mathbf{x}'' \in \Re^c$. Here each component of $\hat{\mathbf{x}}$ is a membership value in some fuzzy set. It appears to us that information possessed by the measured data can be lost in this construction, as the $3p+c$ membership functions used to replace the data are quite subjective. We wonder: when the actual numerical value of a feature is available (even given the possibility of instrumental errors in measurement), why invite more impreciseness? Our point? Authors that augment the SOFM as suggested in [6] and [16] should temper their expectations cautiously—we do not believe there is much to be gained.

References

[1] R. Duda and P. Hart, *Pattern Classification and Scene Analysis.* New York: Wiley-Interscience, 1973.

[2] D. F. Andrews, "Plots of high dimensional data," *Biometrics,* vol. 28, pp. 125–136, 1972.

[3] H. Chernoff, "The use of faces to represent points in k-dimensional space," *J. Amer. Stat. Assoc.*, vol. 68, pp. 361–368, 1973.

[4] B. Kleiner and J. A. Hartigan, "Representing points in many dimensions by trees and castles," *J. Amer. Stat. Assoc.*, vol. 76, pp. 260–269, 1981.

[5] T. Kohonen, *Self-Organization and Associative Memory.* Berlin: Springer-Verlag, 1989.

[6] H. Ritter and T. Kohonen, "Self-organizing semantic maps," *Biol. Cybern.*, vol. 61, pp. 241–254, 1989.

[7] R. A. Johnson and D. W. Wichern, *Applied Multivariate Statistical Analysis.* Englewood Cliffs, NJ: Prentice-Hall, NJ: 1988.

[8] J. W. Sammon, "A nonlinear mapping for data structure analysis," *IEEE Trans. Comput.*, vol. TC-18, pp. 401–409, 1969.

[9] D. H. Foley and J. W. Sammon, "An optimal set of discriminant vectors," *IEEE Trans. Comput.*, vol. 24, pp. 271–278, 1978.

[10] B. Schachter, "A nonlinear mapping algorithm for large data bases," *Comput. Graphics Image Process.*, vol. 7, pp. 271–278, 1978.

[11] C. E. Pykett, "Improving the efficiency of Sammon's nonlinear mapping by using clustering archetypes," *Electron. Lett.*, vol. 14, pp. 799–800, 1980.

[12] C. L. Chang and R. C. T. Lee, "A heuristic relaxation method for nonlinear mapping in cluster analysis," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-3, pp. 197–200, 1973.

[13] G. Biswas, A. K. Jain, and R. C. Dubes, "Evaluation of projection algorithms," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-3, pp. 701–708, 1981.

[14] N. R. Pal, J. C. Bezdek, and E. C. Tsao, "Generalized Clustering networks and Kohonen's self-organizing scheme," *IEEE Trans. Neural Networks.*, vol. 4, no. 4, pp. 549–558, 1993.

[15] J. A. Kangas, T. Kohonen, and J. T. Laaksonen, "Variants of self-organizing maps," *IEEE Trans. Neural Networks.*, vol. 1, no. 1, pp. 93–99, 1990.

[16] S. Mitra and S. K. Pal, "Self-organizing neural network as a fuzzy classifier," *IEEE Trans. Syst., Man, Cybern.*, vol. 24, no. 3, pp. 385–398, 1994.

[17] T. Kohonen, "The self-organized map," *Proc. IEEE,* vol. 78, no. 9, pp. 1464–1480, 1990.

James C. Bezdek (M'80–SM'90–F'92) received the Ph.D. degree from Cornell University in 1973.

He is currently with the Department of Computer Science at the University of West Florida, Pensacola. His interests include pattern recognition, fishing, computational neural networks, skiing, image processing, and medical computing.

Dr. Bezdek is founding editor of the IEEE TRANSACTIONS ON FUZZY SYSTEMS.

Nikhil R. Pal (M'91) received the B.Sc. (Hons.) degree in physics and the M.B.M. degree in operations research in 1979 and 1982, respectively, from the University of Calcutta. He received the M.Tech. and the Ph.D. degrees in computer science from the Indian Statistical Institute, Calcutta, in 1984 and 1991, respectively.

He is an Associate Professor in the Machine Intelligence Unit of the Indian Statistical Institute, Calcutta. He was with the Hindusthan Motors Ltd. W. B., from 1984 to 1985 and with the Dunlop India Ltd., W. B., from 1985 to 1987. In 1987, he joined the Computer Science Unit of Indian Statistical Institute, Calcutta. During Aug. 1991–Feb. 1993 he visited the University of West Florida and is currently also visiting the same university. He was a guest lecturer at the University of Calcutta. His research interests include image processing, pattern recognition, fuzzy sets and systems, uncertainty measures, genetic algorithms, and neural networks.

Dr. Pal is an Associate Editor of the IEEE TRANSACTIONS ON FUZZY SYSTEMS and the *International Journal of Approximate Reasoning.*